Distributed and Anonymous Authentication for unstructured Peer to Peer Networks

- Semester 8 Report -

Pasan Tennakoon
Supipi Karunathilaka

Department of Computer Engineering
University of Peradeniya

Final Year Project (courses CO421 & CO425) report submitted as a requirement of the degree of
B.Sc.Eng. in Computer Engineering

October 2021

Supervisors: Dr. Janaka Alwathugoda (University of Peradeniya)

We would like to dedicate this research to all who helped directly and indirectly to accomplish this. ...

Declaration

We hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my/our own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgments.

Pasan Tennakoon
Supipi Karunathilaka
October 2021

Acknowledgements

We would like to express our deepest appreciation to all those who helped to complete this research. We give our special gratitude to our final year project supervisor, Dr. Janaka Alawatugoda and Prof. Douglas Stebilla who guided us throughout the project.

Abstract

The traditional authentication mechanisms like PKI and ID-PKC are difficult to integrate with a P2P like decentralized network environment. This task becomes even more difficult in an anonymous P2P environment. This research proposes three novel authentication protocols such that users can authenticate themselves in an anonymous P2P network without revealing his/her identity. First, we suggest a way to use existing ring signature schemes to obtain anonymous authentication. Then we propose an anonymous authentication scheme utilizing secret sharing schemes. Finally, we propose a zero-knowledge proof based anonymous authentication protocol. We provide security proofs of the three protocols including anonymity, completeness, soundness, resilience to impersonation and resilient to replay attacks. Then we compare the performance of the new protocols. We deploy these protocols in a P2P environment build using the .Net framework. We utilize Shamir's secret sharing algorithm to manage certificates in the distributed environment.

Table of contents

List of figures

List of tables

Nomenclature

Acronyms / Abbreviations

AS	Authentication Servers
CA	Certificate Authority
DPKI	Distriputed Public Key Infrastructure
FBST	Fair Blind Signature Trust
ID-PKC	Identity based Public Key Certificates
IoV	Internet of Vehicles
MITM	Man In The Middle
NAT	Network Address Translation
P2P	Peer to Peer
PI	Pseudo Identity
PIC	Pseudo Identity Certificate
PKI	Public Key Infrastructure
PPAA	Peer to Peer Anonymous Authentication
PT	Pseudo Trust
RP	Reputed Peers
SP	Super Peers
TOR	The Onion Router

WoT	Web of Trust
ZKP	Zero Knowledge Proof

Chapter 1

Introduction

1.1 Background

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[1], BitTorrent, TOR[2], Freenet[3], 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. Also removes the vulnerability of a single point of failure. P2P networks are considered to be more efficient and scalable than traditional client-server applications.

The decentralized nature of P2P networks makes it difficult to integrate traditional authentication mechanisms. Due to this many such networks focus on providing user anonymity rather than authentication. The reduced security of these networks has created a lot of possible threats [4]. These threats can vary from uploading malicious files to famous Sybil attacks [5]. Also, the anonymity feature of these networks has created a safe house for cybercriminals [6]. Not being accountable for his/her actions, held responsible and punished for malicious actions, P2P users have the freedom to misbehave. This can cause harm to the network and its users.

1.2 The problem

We suggest that even in an anonymous network there should be some level of accountability to protect the network and its users. Accountability is achieved through authentication.

To integrate an authentication mechanism into an anonymous P2P environment we need to solve two main challenges.

- Authenticate in a decentralized environment.
- Authenticate without revealing identity.

These two points have been discussed separately since the start of the internet. Each point has its own difficulties and challenges. Authentication needs to tackle problems like the absence of a central server, certificate management in a distributed environment, the semi-trusted nature of peers, the unpredictable availability of peers, etc. Authentication need to solve problems like not revealing sensitive information about authenticating party's identity, secure against misbehaving parties (cheating verifiers and cheating provers), unlinkability of authentication sessions, practicality, etc.

1.3 The proposed solution

In this paper, we propose three new approaches for anonymous authentication in P2P networks to solve the above problems.

1. Ring signature based approach.
2. Authenticated key sharing based approach.
3. Zero knowledge proof based approach.

We deploy these protocol in a P2P environment where certificates are managed by peers with elevated privileges (super peers). To solve the problems of semi-trusted nature and unpredictable availability of peers we utilize Shamir's secret sharing[7] technique. A more detailed explanation of these cryptographic primitives is given in section 3. Then we test the performance of these ideas in a practical environment developed using the .Net framework.

1.4 Deliverables and milestones

Deliverable	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Week 11	Week 12	Week 13	Week 14	Week 15	Week 16	Week 17	Week 18
Reputation System																		
Anonymous Authentication																		
Distributed Certificate Management																		
Threat Modelling and Performance Evaluation																		

Fig. 1.1 Deliverable and milestones

1.5 Outline of the report

The organization of the paper as follows. In chapter 2 we present the related works of authentication in P2P and anonymous authentication in P2P. Chapter 3 gives a brief introduction to the cryptographic primitives that we used and a high level overview of the entire design including the distributed certificate management system. In chapter 4 we present a descriptive explanation of the proposed protocols and the experimental setup. In chapter 5 we analyze the security of the proposed protocol and the results of the performance testing. In chapter 7 we conclude our research stating possible future works.

Chapter 2

Related work

2.1 Authentication in P2P

Absence of a central server makes authentication in peer to peer (P2P) networks complex. Traditional cryptographic principles like Public Key Infrastructure (PKI) or Identity based Public Key Certificates (ID-PKC) are based on a trusted third party. Establishing a trusted third party in a semi-trusted network like P2P is a questionable task. Many P2P networks propose trust and reputation management schemes to solve this problem.[8] ,[9], [10] 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. There exist plenty of research in this area; EigenTurst [11], NICE[12], Regret [13], PeerTurst[14], FuzzyTrust [15].

P2P systems that use reputation managements schemes to assist in authentication suffer from an obvious flow. 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. For example, CST [**?**] elect a set of peers as RPs (Reputed Peers) using EigenTrust. CST creates a pseudo-identity to hide the real identity of the user. The link between the identity and the pseudo-identity is broken into parts and stored in randomly selected RPs to protect users' privacy. CST trusts RPs to protect the users' identity. However, EigenTrust is vulnerable to collaborative attacks and therefore there exist a possibility that malicious peers are elected as RPs. Malicious RPs can reveal the identity of a user and exploit their privacy.

Some researches suggest using a modified PKI for authentication in P2P networks [16], [17]. Rather than having a single centralized authority, the responsibility of the

Certificate Authority (CA) is distributed across multiple peers in the network. This improves the scalability and robustness of the authentication process.

The downside of using PKI in P2P is certificate management becomes complex. The authentication process becomes difficult to implement effectively. [17] use a set of peers as Authentication Servers (ASs). Even though this improves the scalability of the network, introduce new security risks like unreliability in certificate access and verification.

To solve the problem of the absence of a centralized authority and at the same time keep the authentication process reliable, modern authentication schemes utilize blockchain technology. There is a lot of literature that proposes the idea of using blockchain technology to create a Distributed PKI [18], [19], [20], [21]. This seems to be a good solution to overcome the limitations of having a central trusted certificate authority. Blockchain can make the process of a CA distributed, immutable and transparent. 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.[22] propose the idea of using smart contracts to certificate management.

The DPKI is only secure as long as honest nodes control collectively more than 51% of computing power. Also 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 [23] 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 P2P networks since, 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[24] has been one of the more popular publications of this topic. Pseudo Trust (PT) utilize the concept of double pseudonyms combine with zero knowledge proofs to authenticate users anonymously. PT also uses onion routing[2] and EigenTrust[11] 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. PT neglect one important feature of using the concept of pseudonyms to obtain anonymity. PT does not change PI (pseudo identity) prior to each authentication process. The PT protocol requires PIC (certificate of pseudo identity) to be send to the other party to start the authentication. Since PIC is same for a user, an eavesdropper can link two communication sessions to a specific user.

[25] proposes an similar authentication scheme to PT for Internet of Vehicles (IoV). The only difference is the slight change of the zero knowledge proof and absence of onion routing the trust management. This also suffers from the same vulnerabilities as PT.

[10] present an interesting approach to anonymous authentication. PPAA uses tags to obtain anonymity and at the same time link communication sessions. The idea is to use IDs of the two parties involved in the communication session create a tag. The two parties will not learn any knowledge other than the tag from running the protocol. To avoid having the same tag for different communication sessions between the same parties PPAA propose to include an event id into the tag design. Therefore only a party involved in the communication will be able to link a communication session to a previous session with the same party. The PPAA is secure in random oracle model if eXternal Diffie-Hellmam (XDH) and q-SDH assumption holds.

CST[9] uses collaboration signature to authenticate users anonymously. As mentioned in section 2.1 CST uses EigenTrust[11] reputation system to select trusted peers (RPs). This is not safe in a semi-trusted environment like P2P networks. Other than that CST is said to be resilient against impersonate attacks, traceability and collaboration attacks.

[26] presents a similar method as CST. They use FBST[27][Fair Blind Signatures] to present novel authentication scheme that keep the anonymity of honest users. Similar to 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 trust worthy and play and important role in authentication. However, as mentioned previously trust management systems are not perfect. Malicious peers can get elected as SPs and RPs and they are able to revoke users' anonymity. Similar to CST, [26] uses the concept of Shamir's secret sharing [7] to reduce the vulnerability of exposed RPs. [7] present a way to break a key and store it in multiple places and recreate the key when required. [26] use this technique to break the key (link between ID and pseudo ID) and store it among multiple RPs. Therefore even if few RPs got compromise it does not reveal user's identity. Also a user use anonymous multicast to communicate with a SP. This makes it impossible for a SP to reveal an identity of a user.

[28] uses an combination of Merkle's puzzles[29] and zero knowledge proofs to provide anonymous authentication.

Chapter 3

Methodology

3.1 Cryptographic Primitives

3.1.1 Zero Knowledge Proof

Zero knowledge Proof (ZKP) is a protocol that 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 first idea of ZKP was introduced by Shafi Goldwasser, Silvio Micali, and Charles Rackoff in [30]. Since then many different ZKPs have been published [31], [32], [33], [34]. ZKPs are widely used in cryptography to implement cryptographic protocols due to its privacy, authentication and low complexity.

A zero knowledge proof consists of a prover and verifier. In a zero knowledge protocol, a prover must prove the knowledge of some secret using an interactive challenge-response scheme. The protocol must not reveal any information regarding the secret other than the knowledge of prover has the secret. A secure zkp must satisfy soundness, completeness and zero knowledge properties.

There are two types of zero knowledge systems; interactive zero knowledge proofs and non-interactive zero knowledge proofs [35]. Our proposed protocol uses a zkp that utilize quadratic residues in modular arithmetic.

3.1.2 Ring Signatures

The notion of ring signature was first introduced in 2001 by Ron Rivest, Adi Shamir and Yael Tauman Kalai in [36]. Ring signatures are used to digitally sign messages on behalf of a group. At the same time, makes it computationally difficult to find the exact signer.

Ring signatures are designed to provide anonymity to the message signer. The same functionality is provided by group signatures [37]. The only difference in group signature is that it needs an authoritative entity to generate the signature. Therefore that entity can revoke the anonymity of the signer. Ring signatures do not depend on a third party to generate a signature. Ring signatures are spontaneous and provide unconditional anonymity.

Over the years different ring signature schemes have been published with different features; threshold ring signatures [38], linkable ring signatures[39], revocable ring signatures[40], traceable ring signatures[41].

Consider a scenario where a group of k entities where each entity has a public key Pi and a corresponding secret key Si. An entity 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.

3.1.3 Shamir's Secret Sharing

In 1979 Adi Shamir introduced the concept of Shamir's secret sharing[7][How to Share a Secret]. This allows a secret to be divided into n parts. The secret can be reconstructed with atleast 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 $(a_1, a_2, .., a_{t-1})$. Then set a_0 to the secret we want to share. These points are used to generate the polynomial f(x).

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

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.

$$l_i = \frac{x - x_0}{x_i - x_0} \times \frac{x - x_1}{x_i - x_1} \times ... \times \frac{x - x_{t-1}}{x_i - x_{t-1}}$$

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

The idea of Shamir's secret sharing is a popular concept in p2p systems. A p2p network does not have an centralized database to store peers' keys. Storing keys in a selected set of peers might not be a good idea since p2p is a semi-trusted environment. For an example when a peer request a key from another peer, he might not respond. Therefore keys need to be broken into parts and distributed among multiple peers. A peer should be able to reconstruct a key without the knowledge of all the parts. [9], [26] are p2p anonymous authentication mechanisms that use the concept of Shamir's secret sharing.

3.2 Conceptual design

3.2.1 P2P network design

Using the .Net framework we implemented a hybrid P2P network[42]. A traditional hybrid peer to peer network consists of peers and super peers. Hybrid P2P systems is a combination of purely distributed P2P systems and mediated P2P systems. Hybrid systems are designed to overcome the problems of the two mentioned systems. These systems provide search efficiency of mediated P2P systems while maintaining the reliability of decentralization similar to pure P2P systems[43].

Our P2P network consists of three types of entities; the main server, ordinary peers (hereafter mentioned as peers) and 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 requestors. They are connected to the system through super peers. Every peer is assumed to be behind a NAT environment. Peers with public IP addresses and higher computational power are promoted to be super peers.

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 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 able to change the topology according to this dynamic behaviour of peers and maintain connectivity among existing super peers.

A super peer is only responsible for nodes under his scope and does not know any information regarding other peers of the system. Therefore a node discovery process becomes an exhaustive task. This can be accomplished in two ways; flooding search and

random walk. We utilize flooding search in this project since the random walk is not guaranteed to produce results[44].

3.2.2 Distributed Certificate Management

The decentralized environment of the P2P network does not allow traditional methods of authentication. It's difficult to maintain a centralized database of certificates where the availability of peers cannot be predicted. Distributing certificates among super peers is not a viable solution since super peers are not always available. Therefore all the certificates under this super peer remains not accessible. Also, malicious super peers might delete certificates from the network. The obvious solution is to keep multiple copies of the certificates. We propose a different solution by using Shamir's secret sharing algorithm. The idea is to break the certificates into multiple parts and distribute across the P2P network. When needed, the certificates can be reconstructed from a minimal subset of the parts. A more detailed explanation of the implementation is given in section 3.3.1 .

3.2.3 Proposed Authentication Schemes

Ring Signature Based approach

The characteristics of ring signatures make it an interesting primitive in obtaining anonymous authentication. Ring signatures allows a message to be signed by a group of public keys. Making it impossible to identify the exact signer. The original ring signature scheme[36] and most of the proposed ring signatures provide complete anonymity. This is not suitable for authentication. This make it impossible to revoke the anonymity of malicious peers. Therefore we used the revocable ring signature scheme proposed in [40] to create a simple authentication protocol that protect users' privacy. This is just a simple suggestion, of a way to obtain anonymous authentication using existing ring signature schemes. The idea is to challenge prover to generate a ring signature using a random nonce generated by a verifier. If the prover is able to accomplish this he can successfully authenticate himself.

Authenticated key sharing based approach

We propose a novel authentication mechanism that allows a peer to authenticate without revealing their identity. The basic idea of the protocol is to present prover a set of public keys and challenge to prove the knowledge of atleast one secret key corresponding to

a public key from the set. This idea is simple but the protocol should not reveal any information related to the prover's identity. Also a prover without a valid key pair should not be able to authenticate himself. To accomplish that we employ a authenticated key sharing scheme introduced in [45].

Zero Knowledge Proof Based Approach

Zkp is a popular approach to obtain anonymous authentication in p2p networks. This technique has been utilized in [24], [25] and [10]. Many of these approaches relies on pseudonyms to hide the identity. We propose a new authentication protocol that uses zero knowledge proofs to hide the identity among a group of users. The protocol achieves properties similar to ring signatures. This is an modification of the Schnorr's zero knowledge proof [46]. The method is similar to the authenticated key sharing based approach in the sense that the challenge is to prove the knowledge of a secret key in a set of public keys. However unlike previous method, we use zero knowledge proofs to do that. Therefore this method achieve k anonymity.

3.3 Methodological approach

3.3.1 Distributed Certificate Management

During the initial interaction of 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 these parts across the network. Once a request to recreate the certificate(s) received. Super peer again floods a request(s) to collect the parts of the certificate. The super peers that are holding these parts 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 more dynamic way. As long as r super peers can be accessed, the certificate can be recreated. This method only requires minimal storage. That is, the size of a single part does not exceed the size of the certificate. This is also the more flexible approach. n and r can be changed for each certificate without affecting other certificates. However, then there needs to be a way to identify n and r for each certificate.

n and r are performance metrics. Increasing n will increase the average key storage size in super peers. In section 6 we analyze the performance of increasing n and r.

3.3.2 Proposed Authentication Schemes

Ring Signature Based approach

The protocol starts by the prover collecting a set of certificates from the super peer. Prover then randomly select a subset of the certificates. Then he verify the authenticity of the certificates and obtains the set of public keys from the subset of certificates using main server's public key. Prover hides his own certificate among this subset of certificates and send them to the verifier to initiate the authentication. After authenticating the certificates, verifier obtains the set of public keys using main server's public key. Then verifier generates a random nonce and challenge prover to generate a ring signature for this random nonce, using the above set of public keys. Prover use his secret key, the set of public keys and main server's public key to generate a ring signature according to the algorithm proposed in [40]. Prover then sends the ring signature to the verifier. Verifier verifies the authenticity of the ring signature according to the random nonce he sent at the previous step. If the verification is successful, authentication is complete. Otherwise verifier sends a fail message.

Authenticated key sharing based approach

As same as the previous approach prover collects a set of certificates from the super peer. Then randomly select a subset out of them. After verifying the authenticity of the certificates prover extract the corresponding public keys. Prover mix his certificate into the subset of certificates and send them to the verifier. Verifier obtains the public keys after verifying the authenticity of the certificates. Then generate $X = g^x$ by selecting a random x. Then use the set of public keys to encrypt X. Thus creating a set of ciphertexts where each corresponds to a different public key from the set. Since one of the public key is prover's, he will be able decrypt X with his secret key. After decrypting X, prover selects a random y and calculates $Y = g^y$. Then generate $K = X^y$. K is the shared key. Then he sends Y to the verifier encrypted with verifier's public key. Verifier decrypts Y. Then compute $K = Y^x$. At this stage both parties have the same shared key K. Verifier encrypts a random number R using a symmetric key encryption scheme using K as the key. Then challenge prover to decrypt this and send R back. If the prover generated the correct K at the previous steps, he will be able to decrypt R. Therefore prover can successfully authenticate himself. Otherwise verifier sends a fail message.

Zero Knowledge Proof Based Approach

As same as the above two methods prover collects k certificates from the super peer. Then randomly select n-1 certificates and create C and P vectors as the above methods. However in this method public key is $A_u = g^{a_u} modp$ where a_u is the private key. Similar to Schnorr's protocol prover generates U. The difference is U contains factors of $A_i^{v_i}$ where v_i is a random number. This is generated only using the collected public keys (Prover's public key is not in U). Prover then send U to the verifier. Verifier sends a challenge c to the prover. Prover xor all elements of v_i with c to obtain v_p. Then mix v_p among the set of v_i s and send them along with the set of public keys (including prover's public key) to the verifier. Prover also sends r which is $s - a_p v_p modp$. Then prover does two steps of verification. First he xor v_i s and check if it's equal to c. If it is not terminate the authentication. Otherwise generate U' using r, A and v_i s. If $U = U'$ authentication is successful. Otherwise sends a fail message to the prover. A more detailed explanation is given in section 4.1.3 .

Chapter 4

Experimental Setup and Implementation

4.1 Proposed Schemes

4.1.1 Ring Signature Based approach

Registration

1. A user has an ID which can be anything related to the identity of the user. Selects a random number r_u. Then generate a public key P_u such that

$$P_u = H1(ID, r_u)$$

   User then generates the private key S_u corresponding to P_u

   User sends the registration request along with his ID, 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 the set. After verifying the authenticity of the selected certificates prover generates $C = \{Cert_1, Cert_2, .., Cert_n\}$ which includes prover's certificate $Cert_p$ as well. Prover then obtain each corresponding public key from the

certificates to generate $P = \{P_1, P_2, .., P_n\}$. Then send C to the verifier, encrypted with verifier's public key P_v.

2. Verifier decrypts the message to obtain C. After verifying the authenticity of each $Cert_i$, verifier generates each P_i using main servers public key P_s. Then generate $H = Hash(P)$. Then sends H and a random nonce N to the prover.

3. Prover generate $H' = Hash(P)$ and if $H \neq H'$ terminate the authentication. Otherwise use his secret key S_p, P and P_s to sign N and generate ring signature σ using [40] ring signature scheme. Then send σ to the verifier, encrypted with verifier's public key P_v.

4. Verifier decrypts the message to obtain σ. Then verify whether σ corresponds to N using P set of public keys (obtained in step 2). If the verification is success prover is successfully authenticated. Otherwise verifier sends a fail message.

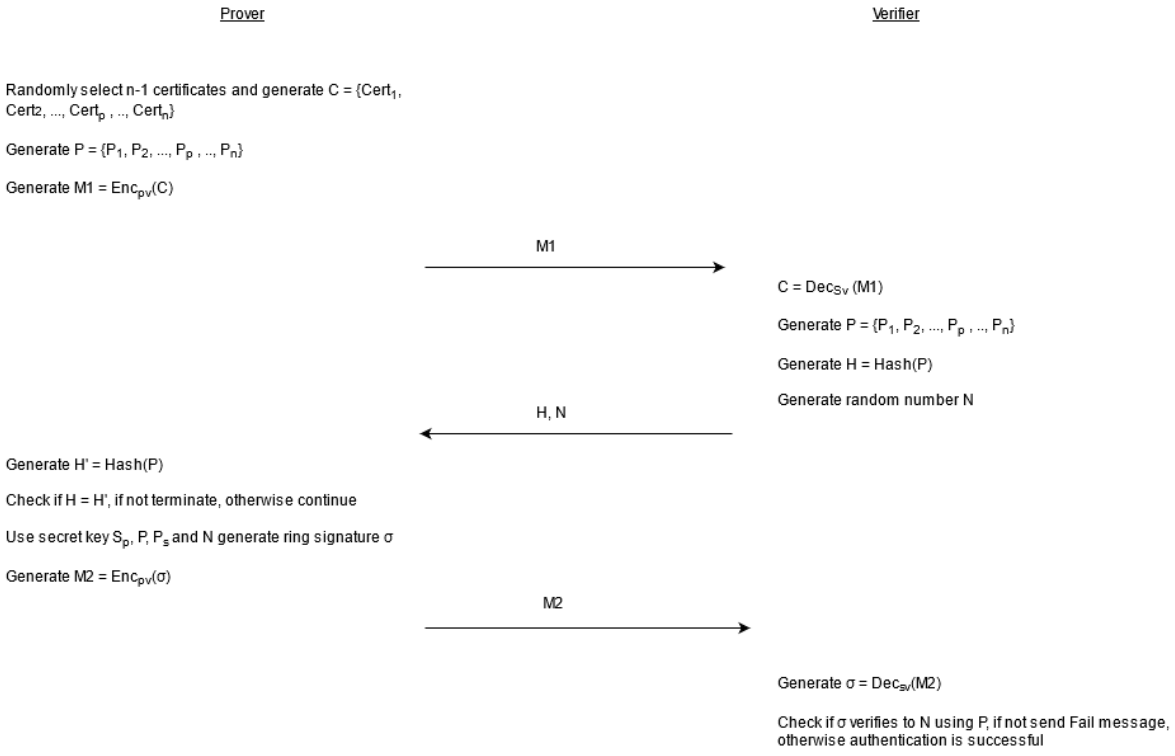

Fig. 4.1 Ring signature based approach authentication

4.1.2 Authenticated key sharing based approach

Registration

1. A user has an ID which can be anything related to the identity of the user. Selects a public r_u. Then generate

$$P_u = H1(ID, r_u)$$

P_u is the public key of the user. User then generates the private key S_u corresponding to P_u

User sends the registration request along with his ID, 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 the set. After verifying the authenticity of the selected certificates prover generates $C = \{Cert_1, Cert_2, .., Cert_n\}$ | C includes $Cert_p$ as well. Then send C to the verifier, encrypted with verifier's public key (P_v).

2. Verifier decrypts P using his secret key (S_v). Generate H = Hash(P). Then generate a random number x and obtain X = g^x. Then generate n ciphertexts $CT = \{C_1, C_2, ..., C_n\} | C_i = E_{P_i}(X|H)$. Verifier sends CT to the prover.

3. Prover selects the C_i corresponding to his public key. Decrypt it using his secret key (S_p) to obtain X and H. Generate H' = Hash(P). Check if $H = H'$. If not terminate the session. Otherwise select a random number y to generate Y = g^y. Then compute K = X^y. Prover sends Y back to the verifier encrypted with P_v.

4. Verifier decrypts Y. Compute K = Y^x. Then generate another random number R, generate $E1_K(R)$. E1(.) is a symmetric key encryption scheme. Then generate $H1 = Hash(R|K)$. Then send $E1_k(R)$ and H1 to the prover.

5. Prover decrypts the message with his knowledge of K to obtain R. Then use R and his K to generate $H1' = Hash(R|K)$. 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. If not verifier sends a fail message to the prover.

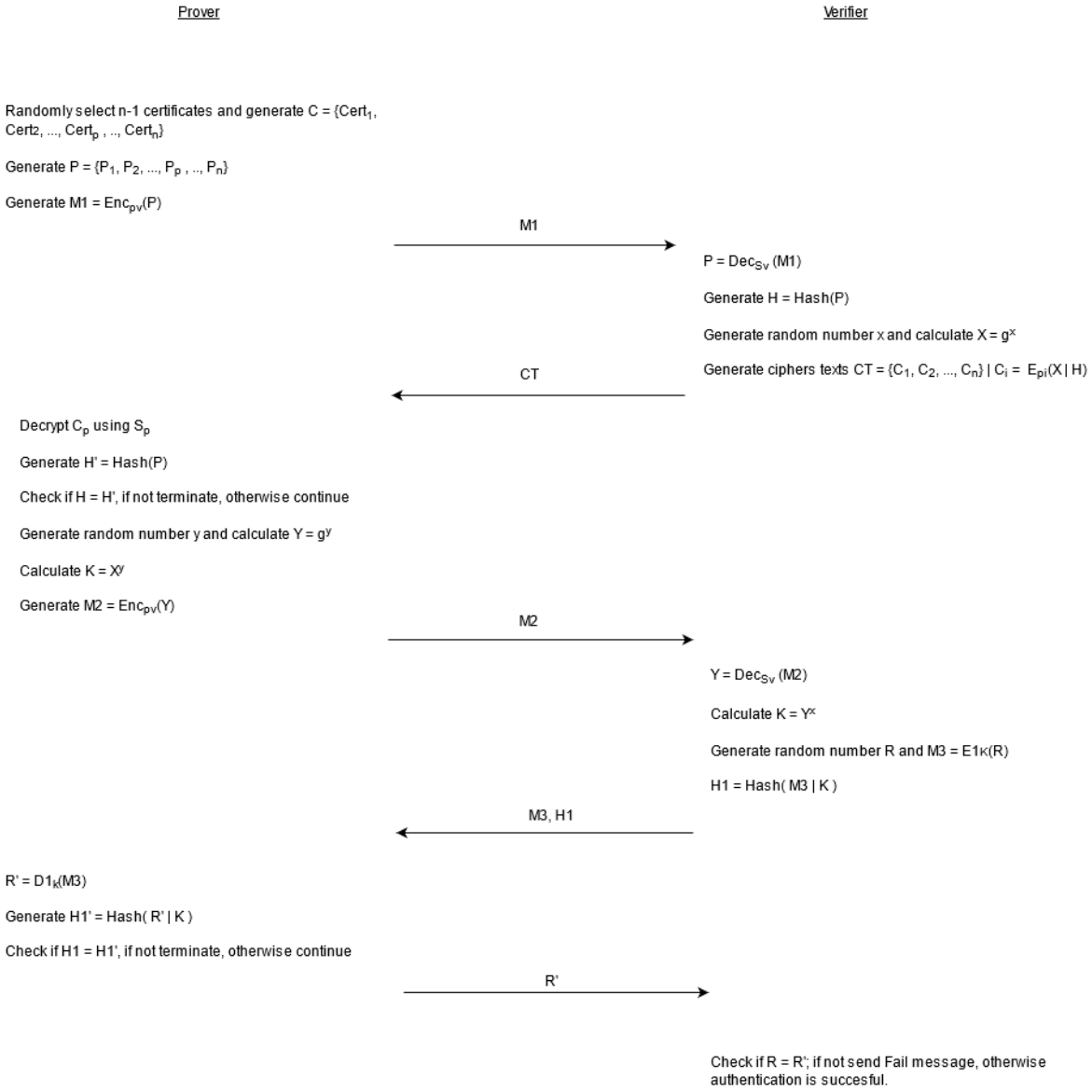

Fig. 4.2 Authenticated key sharing based approach authentication

4.1.3 Zero Knowledge Proof Based Approach

Setup

1. P and Q are two large prime number where P-1 $|Q.g is a generator of a cyclic group of \mathrm{Z}_P^*$ where order of the group is Q. P, Q and g are group parameters.

Registration

1. A user has an ID which can be anything related to the identity of the user. Selects a random integer r_u. Then generate a_u

$$a_u = H1(ID, r_u)$$

such that a_u is from [0, Q-1]. a_u is the private key of the user. Then to generate the public key A_u user calculates

$$A_u = g^{a_u} mod p$$

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 $Cert_u$. 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 the set. After verifying the authenticity of the selected certificates prover generates $C = \{Cert_1, Cert_2, .., Cert_{n-1}\}$. Prover then obtain each corresponding public key from the certificates to generate $P = \{A_1, A_2, \ldots A_{n-1}\}$. Prover then selects a random number s from the range [0, Q-1]. Then selects another n-1 random numbers from the range [0, Q-1] to generate the $V = \{v_1, v_2, \ldots, v_{n-1}\}$. Prover calculates

$$U = g^s A^{v_1} A^{v_2} \ldots A^{v_{n-1}}$$

Prover sends U to the verifier to initiate the authentication.

2. Verifier selects a random number c from the range [0, Q-1] and sends it to the prover.

3. Prover calculates

$$v_p = v_1 \oplus v_2 \oplus \dots v_{n-1} \oplus c$$

Then insert v_p to the vector V such that $V = \{v_1, \dots v_p, \dots v_{n-1}\}$. Prover also update $C = \{Cert_1, ..., Cert_p, ..., Cert_{n-1}\}$ where $Cert_p$ is prover's certificate. Then calculates

$$r = s - a_p v_p mod p$$

Prover sends r, V, C to the verifier.

4. After verifying the authenticity of the certificates in C. Verifier calculates

$$c' = v_1 \oplus v_2 \oplus \dots \oplus v_n$$

If $c \neq c'$, terminate the authentication session. Otherwise calculates

$$U' = g^r A^{v_1} A^{v_2} \dots A^{v_n}$$

If $U = U'$, authentication is successful. Otherwise terminate the authentication.

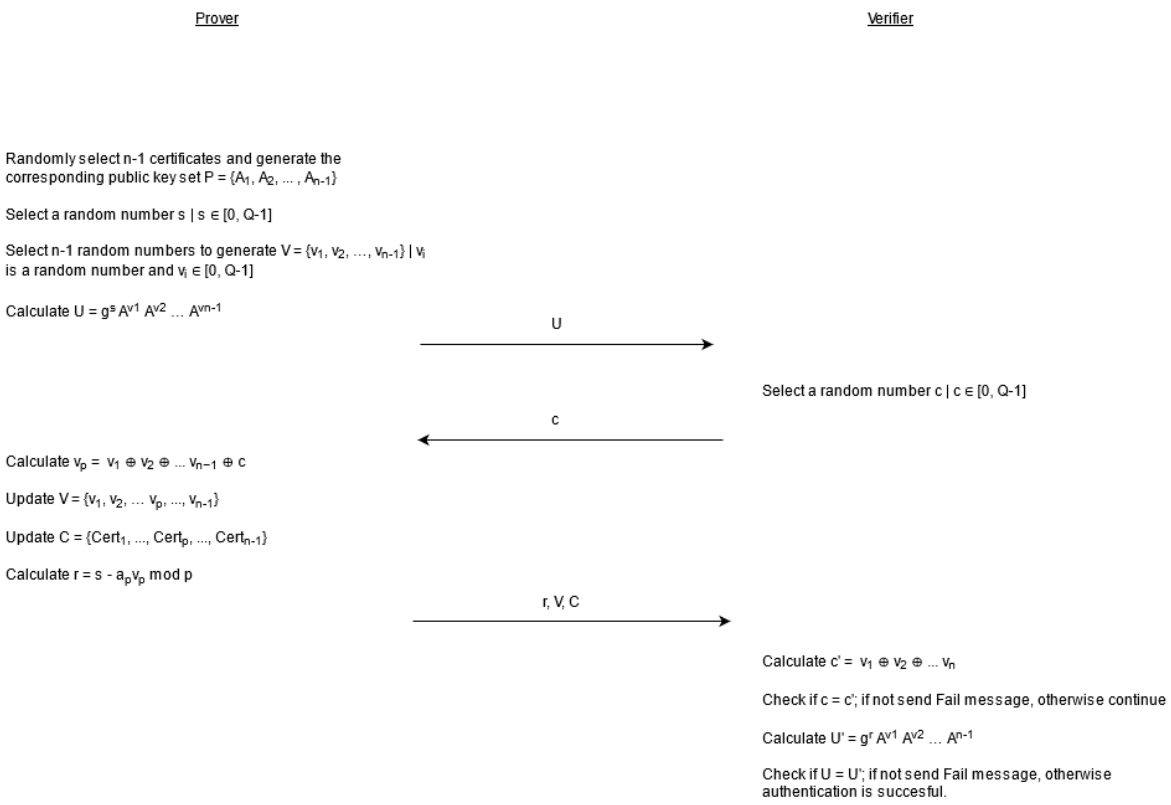

Fig. 4.3 Zero knowledge proof based approach authentication

4.2 Performance Testing

Performance testing of the system was done to understand the capabilities of the system. The testing was done cloud servers located in different countries. Intention was to mimic a world wide distributed network. Although a simulation environment would be ideal to do load testing on the system, absence of open-source platforms to simulate network environments which could run c sharp scripts was a problem. However, a real-world environment helps to understand the system performance in its operating environment. The tests were done to find the limitations of key sharing mechanism and to compare the performance of the three authentication protocols in a real environment.

The first test was done to understand the performance of the key sharing mechanism.

4.2.1 Performance of key sharing

Key sharing technique is an integral part of our implementation. The number of parts the key can be broken into (n) and the number of parts required to reconstruct a certificate

(r) decides the availability of certificates. A high n value and low r value obtains a higher availability. Since distributing the parts of the certificates happens only once, in this experiment we measure the latency of the certificate reconstruction. Specifically, the experiment was done to identify how the latency of a successful certificate reconstruction varies with increasing n and r. The experiment was done by setting n = r. That is all parts of the certificate are required to reconstruct the certificate. First we break a randomly created certificate into n parts and distribute across the P2P network. Then we floods a SEARCH message requesting the parts of the certificate. The time was measured from the time of flooding the SEARCH message until the successful reconstruction of the certificate. We started with breaking the certificate into 2 parts and at each step we increased n by 2. We continued the experiment until n reached 20. For each n, the experiment was done three times and we measured the average time. We also removed any outliers that could affect the results.

4.2.2 Performance of authentication protocols

Anonymity of the authentication protocols depend on the number of certificates. Higher the number of certificates used in the protocol higher the anonymity of the prover. Therefore it is important that a protocol can handle a higher number of certificates. This experiment was done to measure the latency of a complete successful authentication session between a prover and a verifier. At each step we increased the number of ceritificates used in the protocol to measure how the latency varies. The experiment was done for the three proposed protocols in hope to compare the performance. The time was measured from moment that the prover received the requested keys and successful finish of the authentication protocol at the verifiers side. The experiment was started using only 10 keys and at each step we increased the number of keys by 10. The experiment was carried out until 200 keys are used in the authentication protocols. Similar to the previous test, the experiment was done three times and we measured the average time. We also removed any outliers that could affect the results.

Due to limited resources, we used only four publicly available servers each in a different country. The selected servers were located in Singapore, India, America and France. Multiple super-node instances were created at each server and super-nodes were connected so that no two neighbour-nodes reside in the same country. This is to intentionally increase the latency of communication.

Chapter 5

Results and Analysis

5.1 Proofs of security

5.1.1 Ring Signature Based approach

The security of the protocol depends on the security of the ring signature scheme [40]. The authors have proven the correctness, revocation correctness, unforgeability and signer anonymity of the signature scheme. They directly corresponds to the anonymity, completeness, soundness of our suggested protocol.

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 P_p is among the set P. Therefore this obtains k anonymity.

Completeness

If a protocol has completeness, the protocol is said to be comprehensive; an honest verifier will always be able to authenticate himself.

The completeness of the protocol comes from the correctness of the ring signature scheme [40]. The authors of the paper have mathematically proven the correctness of the ring signature scheme. Therefore our protocol is complete.

soundness

If a protocol has soundness property, the protocol is said to be truthful; a cheating prover will never be able to authenticate himself.

Since the ring signature scheme has proven it's unforgeability, a cheating prover will not be able to forge a ring signature. The proposed protocol obtains soundness.

Impersonation

Impersonation is when a malicious user (M) impersonates another user. A protocol that accomplish soundness and completeness is secure against impersonation attacks. Therefore this protocol is secure against impersonation.

Replay Attacks

A replay attack is when an adversary saves a previously sent message(s) and replay it later to gain an advantage. Let's assume a scenario where a malicious user (hereafter mentioned as M) is eavesdropping on a authentication session. M can save message in step 1 (Msg1) and message in step 3 (Msg3), replay it later in the hope to authenticate himself.

Msg1 is encrypted. Therefore M will not be able to reveal it's content. When Msg1 is replayed, verifier will respond with a random N and H. Without the knowledge of P or C prover will not be able to generate the correct ring signature. Therefore will not be able to authenticate himself. Replaying Msg3 will not gain anything unless verifier generates the same N as the original authentication. Probability of this scenario is 1/N, which can be reduced by increasing the domain of N.

5.1.2 Authenticated key sharing based approach

Anonymity

The protocol hides the identity of the prover among a group of selected peers. The group is selected by the prover at random. Therefore verifier cannot manipulate P to obtain a knowledge about the prover.

A cheating verifier may use different x values to obtain prover's identity. Verifier will generate a set of x = $\{x_1, x_2, ..., x_n\}$ and generate X = $\{X_1, X_2, ..., X_n\} \mid X_i = g^{x_i}$. Then verifier can generate CT = $\{C_1, C_2, ..., C_n\} \mid C_i = E_{p_i}(X_i|H)$. By doing so, verifier hope to identify which C_i prover was able to decrypt. Then verifier can link that C_i to corresponding P_i to reveal provers identity.

However, this will not allow verifier to reveal prover's identity since at step 4 verifier needs to generate K without the knowledge of exact X the verifier received. Therefore will not reveal any information about the prover unless verifier can successfully guess the X_i prover decrypted. Successfully random guessing X_i has a probability of 1/n.

Another possibility is using the above method and generating a vector of K = $\{K_1, K_2, ..., K_n\}$ where each K_i correspond to a different x_i. Then at step 4 select a random K_v and send $E1_{k_v}(R)$. By this verifier hopes to find which K_i the prover generated. This can be done by replicating the decryption process using the elements of K vector. Then check what K_i generate a similar output. However this is not possible due to H1 hash. Since this must include the correct key, prover will know the malicious intentions of the verifier and terminate the authentication process.

This methods does not provide k anonymity. Since prover always terminate the authentication whenever the protocol was not correctly followed, verifier can use this knowledge to reduce the scope of prover's identity. For an example, verifier generate CT as half of the C_is are incorrectly formed and other half is correctly formed. If the prover terminate the authentication process, prover's public key is one of the misformed public keys. If the prover continues the authentication process, 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, prover can successfully decrypt X. Therefore can obtain the correct key (k) for step 5.

$$K^{/} = X^{y}$$

$$K^{/} = (g^{x})^{y}$$

$$K^{/} = (g^{y})^{x}$$

$$K^{/} = K$$

Since prover generate the correct key (K). He can successfully decrypt 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 step 3 or

correctly guess R at step 5. Both it is statistically impossible since X and R are generated randomly by the verifier for each communication session.

Therefore unless prover can obtain a secret key and a corresponding public key from a another registered user, it is not possible to authenticate himself.

Impersonation

Since protocol accomplish both soundness and completeness, this protocol is secure against impersonation attacks.

Replay Attacks

A replay attack is when an adversary saves a previously sent message(s) and uses it again to gain an advantage. Let's assume a scenario where a malicious user (hereafter mentioned as M) is eavesdropping on a communication session. M can save message in step 1 (Msg1) , message in step 3 (Msg3) and/or message in step 5 (Msg5), replay it later in the hope 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 random guessing X or R in step 3 and step 5. Storing Msg3 will not help since without the knowledge of y, M will not able to generate K. Only possibility of succeeding in a replay attack is if the verifier generate the same R as the original authentication. Then M can replay Msg5 to successfully authenticate himself as a valid prover.

5.1.3 Zero Knowledge Proof Based Approach

Anonymity

The only information the protocol reveals is that the prover has the knowledge of an a_p. Protocol hides the A_p (public key) corresponds to that a_p among the set of P public keys. Identifying the exact public key of the prover is not feasible. Therefore the protocol obtains k anonymity.

Completeness

If the prover possesses the correct a_p; the secret key corresponding to A_p, only 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 step 1.

$$U' = g^r A^{v_1} \dots A^{v_p} \dots A^{v_{n-1}}$$

$$U' = g^{(s-a_p v_p)} A^{v_1} \dots A^{v_p} \dots A^{v_{n-1}}$$

$$U' = g^s g^{-a_p v_p} A^{v_1} \dots (g^{a_p})^{v_p} \dots A^{v_{n-1}}$$

$$U' = g^s g^{-a_p v_p} A^{v_1} \dots g^{a_p v_p} \dots A^{v_{n-1}}$$

$$U' = g^s A^{v_1} \dots A^{v_{n-1}}$$

$$U' = U$$

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 a a_p a prover will not be able to generate $r = s - a_p v_p mod p$.

At step 3, prover is required to generate v_p by xoring elements of V with the challenge c. This operation ensures that xoring elements in V vector (including v_p) at the verifier's 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. The probability of guessing a_p without any information is 1/Q. Since Q is selected to be a large prime number, probability of that happening is statistically insignificant.

Impersonation

As we explained previously a protocol that accomplish 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 (hereafter mentioned as M) is eavesdropping on a communication session. M can save Msg1 at step 1 and Msg3 at step 3, and replay the messages later in the hope to authenticate himself.

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 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 verifier will terminate the authentication process. This will only be successful if the same c is chosen at the two authentication processes. The probability of this happening is 1/Q. As mentioned in previous cases this is statistically insignificant.

Modifying the Msg3 will not gain any advantage to M. As mentioned under soundness proof, without a valid a_p authenticating will be infeasible.

5.2 Performance Testing

5.2.1 Performance of key sharing

The test was carried out by increasing the number of parts the key is broken into and measuring the latency of a successful key reconstruction.

Number of parts per key	Average Latency (ms)
2	540.26
4	559.08
6	591.55
8	712.93
10	1012.48
12	1120.95
14	1323.59
16	1718.33
18	1404.40
20	4253.17

Table 5.1 Latency of the key reconstruction with increasing number of parts per key.

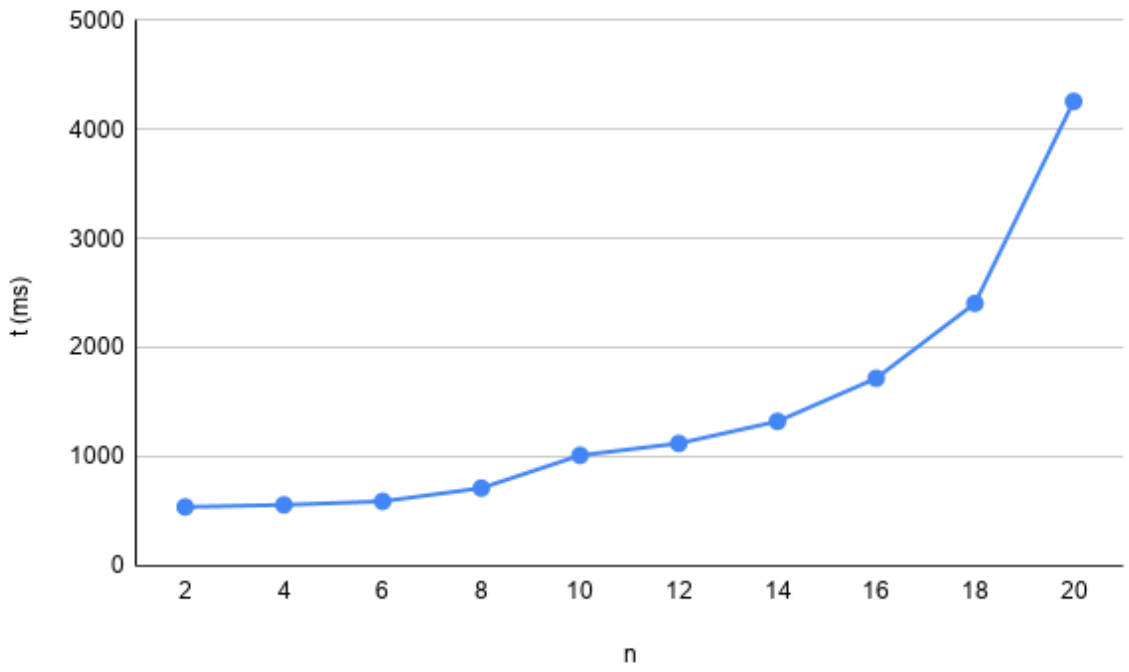

Fig. 5.1 Performance of the key reconstruction with increasing number of key parts

5.2.2 Performance of authentication protocols

The test was carried out by increasing the number of keys used in the authentication process and measuring the latency of a successful authentication process. The test was carried out separately for the three authentication protocols in similar environments.

Number of Keys	Latency in Authenticated key sharing based approach (ms)	Latency in Zero knowledge proof based approach (ms)	Latency in Ring signature based approach (ms)
10	82.00	58.33	98.67
20	87.00	56.33	107.00
30	88.00	62.00	104.00
40	89.33	80.00	106.67
50	99.67	58.00	114.67
60	99.67	56.33	109.33
70	105.00	58.67	119.67
80	111.33	68.33	121.33
90	112.67	63.67	127.33
100	116.00	57.67	120.67
110	105.33	59.33	124.33
120	115.00	64.33	127.33
130	119.67	55.67	142.67
140	124.67	58.67	144.33
150	127.33	56.33	144.67
160	113.33	67.33	138.67
170	134.33	58.67	140.00
180	128.00	58.33	144.00
190	125.67	56.00	142.00
200	133.00	63.33	144.00

Table 5.2 Latency of the authentication protocols with increasing number of keys.

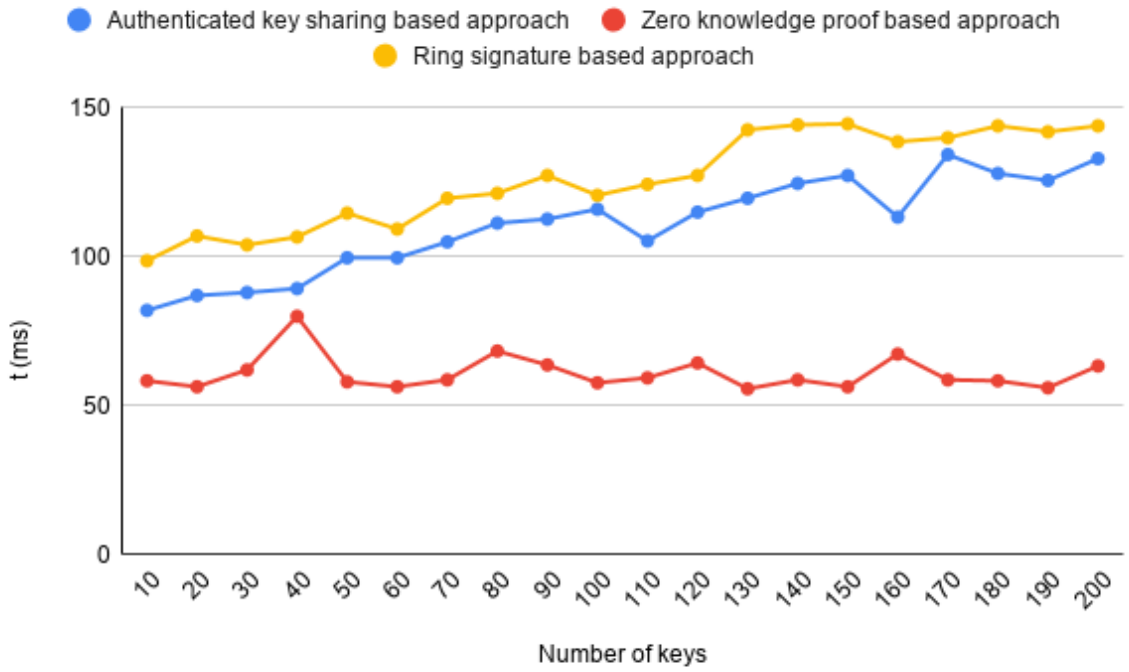

Fig. 5.2 Performance comparison of the three authentication protocols

5.3 Performance Analysis

According to Fig. 5.1 the latency of the key reconstruction exponentially increases with increasing number of key parts. The maximum number of possible parts per key without increasing a threshold of 5 seconds is 20. The reason for such exponential increment is due to the linear distribution of unique keys. That is to request the m^th part of the key, the request need to travel through m super nodes. Increasing the number of parts per key will increase the overall distance the request message has to travel through. Increasing the number of key parts increases the overall latency of the authentication process. Higher the parts the key is broken to, higher the availability of the keys. Therefore need to understand the optimal trade off between the availability of keys and the latency of the key reconstruction process.

It is also important to notice that the network congestion at the time of the experiment, the physical location of super nodes, the performance of the super nodes can have an effect on the results. Nonetheless, the experiment is still essential to get an overall idea of the performance of the key reconstruction mechanism.

The Fig 5.2 compare the performance of the three authentication protocols with respect to time. The latency of the authenticated key sharing based approach and the ring signature based approach increases with the increasing number of keys. However, the latency of the zero knowledge proof based approach remains constant. The reason is the encryption and decryption steps of the first two protocols. Higher the number of keys, higher the number of encryptions and decryptions the protocol has to done. The zero knowledge proof based approach has no explicit encryption and decryption steps. Thus it is faster and no visible increment in the latency with increasing number of keys compared to the other two approaches.

Higher the number of keys used in the authentication process higher the anonymity. Since, the zero knowledge proof based approach can increase the number of keys used in the protocol without increasing the latency of the authentication process it is more efficient compared to the other two protocols.

Chapter 6

Conclusions and Future Works

We have proposed three novel protocols to achieve anonymous authentication in peer to peer networks. The ring signature-based approach provides a suggestion of how to utilized already implemented ring signatures to obtain anonymous authentication. Then we propose an authentication method that utilizes a key sharing mechanism. This method does not provide zero-knowledge. That is a verifier can obtain some knowledge of the prover identity. To solve this we introduce a zero-knowledge proof-based approach that utilizes Schnorr's protocol to achieve anonymous authentication. This method is more efficient, secure and most importantly achieve zero knowledge. We have proven the security of each protocol including anonymity, completeness, soundness, resilience to impersonation and resilience to replay attacks. The protocols were tested in a peer to peer overlay network build using the .Net framework. The peer to peer network utilizes a distributed certificate management mechanism build using Shamir's secret sharing algorithm. This allows us to access certificates in a more efficient manner compared to the traditional approaches.

As for future works, we hope to modify the zero-knowledge proof-based approach for certificate revocation. That is, give an authoritative entity the privilege to revoke the anonymity of a user when required. We also hope to integrate the proposed authentication protocols in real-world peer to peer transactions.

References

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

[2] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," *Paul Syverson*, vol. 13, 06 2004.

[3] I. Clarke, O. Sandberg, B. Wiley, and T. 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*, pp. 46–66, Springer-Verlag New York, Inc., 2001.

[4] D. Wallach, "A survey of peer-to-peer security issues," vol. 2609, pp. 42–57, 01 2002.

[5] J. R. Douceur, "The sybil attack," in *Revised Papers from the First International Workshop on Peer-to-Peer Systems*, IPTPS '01, (Berlin, Heidelberg), p. 251–260, Springer-Verlag, 2002.

[6] E. Jardine, "The dark web dilemma: Tor, anonymity and online policing," *Global Commission on Internet Governance Paper Series*, p. 24, 09 2015.

[7] A. Shamir, "How to share a secret."

[8] S. Gokhale and P. Dasgupta, "Distributed authentication for https://www.overleaf.com/project/601cbf829bf80103a1bd0b4dpeer-to-peer networks," pp. 347– 353, 02 2003.

[9] X. Wang, S. Xingming, G. Sun, and D. Luo, "Cst: P2p anonymous authentication system based on collaboration signature," *2010 5th International Conference on Future Information Technology, FutureTech 2010 - Proceedings*, 01 2010.

[10] P. Tsang and S. Smith, "Ppaa: Peer-to-peer anonymous authentication," pp. 55–74, 06 2008.

[11] S. Kamvar, M. Schlosser, and H. Garcia-molina, "The eigentrust algorithm for reputation management in p2p networks," *The EigenTrust Algorithm for Reputation Management in P2P Networks*, 04 2003.

[12] S. Lee, R. Sherwood, and B. Bhattacharjee, "Cooperative peer groups in nice," vol. 50, pp. 1272 – 1282 vol.2, 03 2006.

[13] J. Sabater-Mir and C. Sierra, "Reputation and social network analysis in multi-agent systems," pp. 475–482, 01 2002.

[14] L. Xiong and L. Liu, "A reputation-based trust model for peer-to-peer ecommerce communities," pp. 275 – 284, 07 2003.

[15] S. Song, K. Hwang, R. Zhou, and Y.-K. Kwok, "Trusted p2p transactions with fuzzy reputation aggregation," *Internet Computing, IEEE*, vol. 9, pp. 24 – 34, 12 2005.

[16] B.-T. Oh, S.-B. Lee, and H.-J. Park, "A peer mutual authentication method using pki on super peer based peer-to-peer systems," vol. 3, pp. 2221 – 2225, 03 2008.

[17] W. Josephson, E. Sirer, and F. Schneider, "Peer-to-peer authentication with a," 03 2004.

[18] A. Papageorgiou, A. Mygiakis, K. Loupos, and T. Krousarlis, "Dpki: A blockchain-based decentralized public key infrastructure system," pp. 1–5, 06 2020.

[19] E. Karaarslan and E. Adiguzel, "Blockchain based dns and pki solutions," *IEEE Communications Standards Magazine*, vol. 2, pp. 52–57, 09 2018.

[20] A. Yakubov, W. Shbair, A. Wallbom, D. Sanda, and R. State, "A blockchain-based pki management framework," 04 2018.

[21] H. Orman, "Blockchain: The emperors new pki?," *IEEE Internet Computing*, vol. 22, pp. 23–28, 03 2018.

[22] P. Sivakumar and K. Singh, "Privacy based decentralized Public Key Infrastructure implementation using Smart contract in Blockchain," *2nd Advanced Workshop on Blockchain: Technology, Applications, Challenges*, vol. 2, no. 1, pp. 1–6, 2017.

[23] A. Bob, A. David, B. Greg, and E. Fred, "The pgp trust model," 2005.

[24] L. Lu, J. Han, L. Hu, J. Huai, Y. Liu, and L. Ni, "Pseudo trust: Zero-knowledge based authentication in anonymous peer-to-peer protocols," pp. 1 – 10, 04 2007.

[25] M. Han, Z. Yin, P. Cheng, X. Zhang, and S. Ma, "Zero-knowledge identity authentication for internet of vehicles: Improvement and application," *PLoS ONE*, vol. 15, 9 2020.

[26] "Fair blind signature based authentication for super peer p2p network(2),"

[27] P. Jean-Marc, "Fair blind signatures," 2000.

[28] A. Wierzbicki, A. Zwierko, and Z. Kotulski, "Authentication with controlled anonymity in p2p systems," 2005.

[29] S. Pasini, "Secure communications over insecure channels using an authenticated channel," 01 2005.

[30] S. Goldwasser, S. Micali, and C. Rackoff, *The knowledge complexity of interactive proof-systems*. 10 2019.

[31] C. Tang, Z. Liu, and J. Liu, "The statistical zero-knowledge proof for blum integer based on discrete logarithm," 12 2003.

[32] U. Feige, A. Fiat, and A. Shamir, "Zero-knowledge proofs of identity," *Journal of Cryptology*, vol. 1, pp. 77–94, 06 1988.

[33] R. Cramer and I. Damgård, "Linear zero-knowledge – a note on efficient zero-knowledge proofs and arguments," vol. 3, pp. 436–445, 05 1997.

[34] A. Sahai and S. Vadhan, "A complete problem for statistical zero knowledge.," *IACR Cryptology ePrint Archive*, vol. 2000, p. 56, 01 2000.

[35] H. Wu and F. Wang, "A survey of noninteractive zero knowledge proof system and its applications," *TheScientificWorldJournal*, vol. 2014, p. 560484, 05 2014.

[36] R. Rivest, A. Shamir, and Y. Tauman, "How to leak a secret," vol. 2248, pp. 552–565, 07 2001.

[37] D. W. Davies, "Advances in Cryptology–EUROCRYPT '91: Workshop on the theory and application of cryptographic techniques Brighton, UK, April 8-11, 199 1 proceedings," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 547 LNCS, no. iii, pp. 257–265, 1991.

[38] E. Bresson, J. Stern, and M. Szydlo, "Threshold ring signatures and applications to ad-hoc groups," vol. 2442, pp. 465–480, 08 2002.

[39] J. Liu and D. Wong, "Linkable ring signatures: Security models and new schemes," vol. 3481, pp. 614–623, 05 2005.

[40] D. Liu, J. Liu, Y. Mu, W. Susilo, and D. Wong, "Revocable ring signature," *J. Comput. Sci. Technol.*, vol. 22, pp. 785–794, 11 2007.

[41] E. Fujisaki and K. Suzuki, "Traceable ring signature," vol. E91.A, pp. 181–200, 06 2007.

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

[43] P. Backx, T. Wauters, B. Dhoedt, and P. Demeester, "A comparison of peer-to-peer architectures A comparison of peer-to-peer architectures," no. January, 2002.

[44] R. Ahmed and R. Boutaba, "A survey of distributed search techniques in large scale distributed systems," *IEEE Communications Surveys and Tutorials*, vol. 13, no. 2, pp. 150–167, 2011.

[45] J. Alawatugoda, "Generic construction of an eck-secure key exchange protocol in the standard model," *Int. J. Inf. Secur.*, vol. 16, p. 541–557, Oct. 2017.

[46] A. P. Soares, "Schnorr Non-interactive Zero-Knowledge Proof," *Journal of Chemical Information and Modeling*, vol. 53, no. 9, pp. 1689–1699, 2013.