A STUDY ON USE OF ZERO KNOWLEDGE AUTHENTICATIONS AT P2P PROTOCOLS

Sharath Yaji.B.E.M.Tech

Asst.Professor
Dept.of Computer Science
Canara Engineering College. Benjanpadavu
sharath_yaji@yahoo.com

ABSTRACT

The whole point of cryptography is to solve problems. Cryptography solves problems that involve secrecy, authentication, integrity, and dishonest people. A cryptographic protocol is a protocol that uses cryptography. Cryptography is an art and science of keeping messages secure. A cryptographic protocol involves some cryptographic algorithm, but generally the goal of the protocol is something beyond simple secrecy.

Zero knowledge authentication is an advanced cryptographic authentication. This is an interactive authentication between prover and verifier. The prover convinces her knowledge to Verifier and the verifier verifies it. This is the strategy using in zero knowledge authentication.

At present peer-to-peer (P2P) systems are identity based, which means that in order for one peer to trust another, it needs to know the other Peer's identity. Hence, there exists an inherent trade off between trust and anonymity. A new authentication scheme based on Zero-Knowledge authentication is designed so that peers can be authenticated without leaking any sensitive information which is named Pseudo Trust(PT). The security of most of the zero-knowledge proof of identity protocols is based on complex mathematical algorithms and requires heavy computations for both parties involved, the prover and the verifier. Thus, the two parties must depend on computing devices (computers) to perform these computations.[1]

1. INTRODUCTION

A protocol is a series of steps, involving two or more parties, designed to accomplish a task. The parties participating in the protocol might want to share parts of their secrets to compute a value, jointly generate a random sequence, convince one another of their identity, or simultaneously sign a contract. The whole point of using cryptography in a protocol is to prevent or detect eavesdropping and cheating. In daily life, there are informal protocols for almost everything: ordering goods over the telephone, playing poker, voting in an election. No one thinks much about these protocols; they have evolved over time, everyone knows how to use them, and they work reasonably well. These days, more and more human interaction takes place over computer networks instead of face-to-face. Computers need formal protocols to do the same things that people do without thinking. Many face-to- face protocols rely on people's presence to ensure fairness and security.

Zero knowledge authentication is an interactive authentication method in prover proves the knowledge and verifier verify it. Zero-knowledge protocols allow identification, key exchange and other basic cryptographic operations to be implemented without leaking any secret information during the conversation and with smaller computational requirements than using comparable public key protocols. Thus Zero-knowledge protocols seem very attractive especially in smart card and embedded applications.

As an emerging model of communication and computation, peer-to-peer (P2P) networking has recently gained significant acceptance. Millions of users share huge amounts of resources by forming an abstract, logical network called an overlay network. Most widely-deployed P2P systems today, including Gnutella, KaZaA, and BitTorrent, employ a routed-search-and-direct-download mechanism. Peers are linked in the overlay network, each maintaining several logical neighbours. Query flooding is the most popular search method in such systems. If a peer receiving a query can provide the requested object, a response message is sent back to the requesting peer, and a direct download path is constructed between the downloader and the content provider. One drawback of the above protocols is the fact that such P2P systems might compromise users' privacy. The IP addresses of object requesters and providers can easily be discovered and translated into users' names and postal addresses. Hence, many studies such as the Peer-to-Peer Personal Privacy Protocol (P5) and Anonymous Peerto-Peer File Sharing (APFS) focus on providing anonymous searching and downloading in P2P systems.[1].

The purpose of designing an anonymous authentication protocol in P2P systems is motivated by a specific *problem: how to support authentication without exposing the real identities of peers*. IN Pseudo Trust (PT) protocol, in which each peer generates an unforgivable and verifiable pseudonym using a one-way hash function. Such one-way mapping can effectively defend against impersonation, forgery, and Man-In-Middle-Attacks, so that the pseudonyms can be used as the real IDs in P2P systems. This means that previous methods of identity-based trust management can be adopted. Thus, trust management can be pseudonym based so that the real identities of peers are protected and users are able to verify each other without leaking any sensitive or private information. The salient features of Pseudo Trust include (1) achieving anonymity as well as authentication, (2) eliminating the support of a centralized CA system, and (3) resisting man-in-middle-attacks.

Peer-to-peer (P2P) networks are very popular in today's communicating, especially in the research of cooperative and parallel computation, the shared data of E-business and civil industry. One of key requirements for the control mechanism in P2P networks is access control which becomes more critical when Peers enter into or leave a collaboration community. However, the conventional access control mechanism is not suitable because the P2P networks is decentralize and dynamic one . [2] The first party in this protocol, the prover, wants to identify himself or herself by proving that he or she possesses a certain piece of secret information to the second party, the verifier, without revealing the secret information itself to the verifier or to any third party during the proof process .For

example, Alice, the prover, has a secret and wants to prove her knowledge of the secret to Bob, the verifier, without disclosing the secret itself. The zero-knowledge proof of identity has to satisfy three properties :

a. *Completeness property:* If the statement is true, the real prover can convince the real verifier that it is true.

b. *Soundness property*: If the statement is false, the cheating prover cannot convince the real verifier that it is true.

c. *Zero-knowledge property*: If the statement is true, the cheating verifier cannot learn anything other than the public data of the real prover.

The security of most zero-knowledge proof of identity protocols is based on complex mathematical algorithms and is required heavy computations for both parties involved, the prover and the verifier. Thus, the two parties must depend on computing devices (computers) to perform these computations. From this point of view, we work to create a

new protocol for zero-knowledge proof of identity that does not require complex mathematical computations, i.e. a new protocol with a comparatively low and simple computation complexity and without the need for any special tools. Table 1 compares, in summary, the requirements of different cryptographic systems [3].

Table 1. Brief comparison among the zero-knowledge, the public-key, and the secret-key systems

System name	Message size	System iteration	Amount of computation	Memory requirements
Zero-knowledge	Large	Many	Large	large
Public-key	Large	One	Very large	large
Secret-key	Small	One	Small	Small

2.Zero-knowledge Protocol Basics

Zero-knowledge protocols, as their name says, are cryptographic protocols which do not reveal the information or secret itself during the protocol, or to any eavesdropper. They have some very interesting properties, e.g. as the secret itself (e.g. your identity) is not transferred to the verifying party, they cannot try to masquerade as you to any third party. Although Zero-knowledge protocols look a bit unusual, most usual cryptographic problems

can be solved by using them, as well as with public key cryptography. For some applications, like key exchange (for later normal cheap and fast symmetric encryption on the communications link) or proving mutual identities, zero-knowledge protocols can in

many occasions be a very good and suitable solution.The following *people* appear in zero-knowledge protocols:

Peggy the Prover

Peggy has some information that she wants to prove to Victor, but she doesn't want to tell the secret itself to Victor.

Victor the Verifier

Victor asks Peggy a series of questions, trying to find out if Peggy really knows the secret or not.Victor does not learn anything of the secret itself, even if he would cheat or not adhere to the protocol.

Eve the Eavesdropper

Eve is listening to the conversation between Peggy and Victor. A good zero-knowledge protocol also makes sure that any third-party will not learn a thing about the secret, and will not even be able to replay it for anyone else later to convince them.

Maggie the Malice

Maggie is listening to the protocol traffic and maliciously sending extra messages and modifying or destroying messages. The protocol must be tamper-resistant to this kind of activity.

Let, There is a circular cave with an entrance on one side and a magic door inside the cave

Blocking the opposite side. The first party, the prover, Peggy, possesses the secret word to open the magic door in the circular cave and wants to prove to second party, the verifier, Victor, that she really knows the secret word that can open the magic door, without revealing the secret itself. The two parties, according to the analogy, proceed through the following steps (see Fig. 1) :

1) Peggy goes into the cave to the branching point of the two paths (A and B) while Victor waits outside.

2) Peggy randomly selects path A or B and goes to the magic door while Victor goes into the cave to the branching point of the two paths.

3) Victor calls out a random path of the cave (A or B), where Peggy should come out.

4) Peggy can come out from the correct path every time if she knows the secret word to open the door, as she can open and pass the magic door with the secret word if necessary. If she doesn't have the secret word, she has a 50% probability of coming out from the wrong path because she cannot pass the magic door. In the latter case, Victor will call her on a cheat the cheater. Peggy and Victor repeat these steps as many times as required to convince Victor. If, after many repetitions, Peggy comes out from the correct path each time, Victor can be convinced that Peggy possesses the secret word to open the magic door.[3]

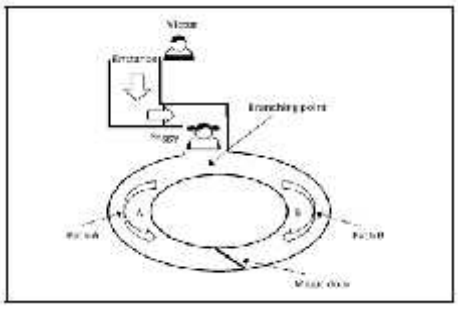

Fig1 : Zero-knowledge (Ali Baba's) cave

2.1 Zero-knowledge terminology:

The *secret* means some piece of information, be it a password, the private key of a public key cryptosystem, a solution to some mathematical problem or a set of credentials.With Zero-knowledge protocols, the prover can convince the verifier that she is in possession of the knowledge, the secret, without revealing the secret itself, unlike e.g.normal username-password queries.

Accreditation means the building of confidence in each iteration of the protocol. If in one

step of a zero-knowledge protocol, the chance of an impostor being able to provide the answer is 1 in 2, the chances of her passing an entire conversation are 2^-(number of

accreditation rounds).Often the prover will offer a *problem* (i.e. particular numeric values for a generic hard-to solve mathematical problem, e.g. factoring extremely large numbers, which are products of large primes) to the verifier, which will ask for one of the 2 or more possible solutions. If the prover knows the real solution to the hard mathematical problem, she is able to provide any of the solutions asked for. If she doesn't know the real solution, she cannot provide all of the possible solutions, and if the verifier asks for one of the other solutions, she is unable to provide it, and will be found out.

Cut-and-choose protocols work in the way, that one failure means the failure of the whole

protocol (i.e. that the prover is not legitimate), but you can keep working on the protocol as

long as you want, if the prover is legitimate. After you reach the level of confidence you
need without being cut off, the protocol is successful.

2.2 Features of Zero-knowledge Protocols:

Zero-knowledge protocols can be described as cryptographic protocols having the following special features:

The verifier can not learn anything from the protocol :

The Verifier does not learn anything from the protocol, that he could not learn all by himself, without the prover. This is the central zero-knowledge concept, i.e. No knowledge is transferred. Without this feature, the protocol would be called a minimum-disclosure protocol, i.e. zero-knowledge protocols require that absolutely no information can be leaked in any case.

The prover cannot cheat the verifier:

If Peggy doesn't know the secret, she can only succeed with a great amount of good luck. After several rounds of the protocol, the odds of an impostor passing as legitimate can be made as low as necessary. The protocols are also cut and choose, i.e. the first time the prover fails, Victor knows Peggy is not legitimate. So, with each round of the protocol, the certainty gets better and better. The protocols can be made to work even if the odds of a guess passing are high, you just need more rounds in the protocol.

The verifier can't cheat the prover :

Victor can't get any information out of the protocol, even if he does not follow the protocol. The only thing Victor can do is to convince himself that Peggy knows the secret. The Prover will always reveal only one solution of many to any one problem, never all of them which would allow finding out the secret itself.

The verifier can not pretend to be the prover to any third party:

Because no information can leak from Peggy to Victor, Victor can't try to as querade as Peggy to any outside third party. With some of these protocols, a man-in-the middle attack is possible, though, meaning that someone can relay the traffic from the true Peggy and try to convince another Victor that he, the perpetrator, is Peggy. Also, if the verifier records the conversation between him and the prover, that recording can't be used to convince any third party. It looks the same as a faked conversation (e.g. where the verifier and prover agreed beforehand which requests the verifier will choose).

2.3 Modes of operations:

The zero-knowledge protocols can be used in three main modes.

Interactive, where Peggy and Victor interactively go through the protocol, building up
the certainty piece by piece.

Parallel, where Peggy creates a number of problems and Victor asks for a number of solutions at a time. This can be used to bring down the number of interactive messages with
a slow-response-time connection.

Off line, where Peggy creates a number of problems, and then uses a cryptographically strong one-way hash function on the data and the set of problems to play the role of Victor,
to select a random solution wanted for each problem. She then appends these solutions to
the message. This mode can be used for *digital signatures.*

2.4 System Requirements:

Many sources claim that Zero-Knowledge protocols have lighter computational requirements than e.g. public key protocols. The usual claim is that Zero-Knowledge Protocols can achieve the same results than public key protocols with one to two orders of Magnitude less (1/10 1/100) computing power. A typical implementation might require 20 30 modular multiplications (with full length bit strings) that can be optimized to 10 20 with pre calculation. This is much faster than RSA. The memory requirements seem to be about equal - to have very high security with Zero-knowledge protocols, you will need very long keys and numbers, so in memory terms, the requirements may not be very different.

3 RELATED WORKS

This section, describes related works in authentication, anonymity, ZKP and PAKE.

3.1 Peer-to-Peer Trust and Authentication Schemes

Abdul-Rahman and Halles propose a trust model and a recommendation protocol, focusing on decentralized systems. Aprimeand clear definition of trust is also provided. XREP enables peers to evaluate and share other peer reputations by introducing a distributed polling algorithm. XREP also employs confirmation voting procedures among randomly chosen peers in order to prevent collusive cheating from cliques of malicious peers. The P-Grid focuses on an efficient data management technique to construct a scalable trust model for decentralized applications. EigenTrus builds a virtual global matrix to represent individual reputations. NICE provides a platform to implement distributed cooperative applications. Based on trust chains, NICE computes a user reputation in a PGP-like model. XenoTrust provides a public infrastructure for a wide area trust computing environment. Generally, most P2P trust designs are identity based, where one peer does not trust another before knowing its identity.

3.2 Anonymity

Privacy has become an increasingly salient issue, and considerable progress has been made with anonymous communications. Several solutions achieve mutual anonymity for both initiators and responders in P2P systems, which generally aim at concealing the real identities of users during transactions. For example, in APFS, peers construct an anonymous path with tail nodes using an onion technique, providing complete and mutual anonymity for peers. Recent research has attempted to introduce reputation value into anonymous P2P systems or construct a trust management based on proxy techniques. However, failure to support authentication makes these approaches vulnerable to impersonation and MIMAs. Therefore, it is argued that providing privacy to peers increases the difficulties of authenticity and security. Obviously, there is a trade-off between authentication and anonymity.

3.3 Zero-Knowledge Proof

The purpose of ZKP protocols is to help a prover convince a verifier that she holds some knowledge (usually secret),without leaking any information about the knowledge during the verification process (zero-knowledge). The concept of ZKP was first introduced by Goldwasser and has since been employed in many authentication and identification protocols. Loosely speaking, a ZKP is an interactive proof system, which is comprised of a prover and a verifier. The principle rule is that the prover demonstrates knowledge of a secret to the verifier through several interactive rounds. During the process, the prover does not reveal any sensitive information to the verifier or any other parties. Each round involves a challenge (say, a question) from the verifier and a response (say, an answer) from the prover. If the secrets are related to user identities, ZKP can be used for identification and, in this case, is called Zero-Knowledge Proof of Identity (ZKPI). The security of ZKPI protocols is often based on the intractability of factoring large integers or computing a discrete logarithm problem. Some have been improved to employ mutual authentication and key exchanges . However, since almost all ZKP-based identification schemes are dependent on a trusted third party (such as a CA) as an authorized central server, they are not directly adopted by this design.[4]

3.4 PAKE

Password-Authenticated Key Exchange (PAKE) is a family of protocols that affords a reasonable level of security using short memorized passwords for protecting information over insecure

channels. Such protocols are also a topic of IEEE P13637 standard working group. Encrypted Key Exchange (EKE) that was introduced in 1992, combines both asymmetric and symmetric cryptography findings. The protocol has several versions and was followed by other propositions. Simplified Password authenticated Exponential Key Exchange (SPEKE) protocol was developed by Jablon [5] for commercial purposes.The main difference in comparison with EKE is that the password is used to influence the selection of the generator parameter in the session-key generation function. Secure Remote Password (SRP)8 [5] was developed in 1997. Security of this protocol is dependent on the strength of the applied one-way hash function. The protocol was revised several times, and is currently at revision six. SRP is often applied to telnet and ftp. The protocol is more computationally intensive than EKE. It requires two modulo exponentiations, whereas EKE requires only one. Moreover, the protocol is vulnerable to offline dictionary attacks.

4. PSEUDO TRUST

The real and specific challenge that underlies the tradeoff between trust and anonymity is that on one hand, all existing P2P trust systems attempt to link each peer ID with a trust value; on the other hand, anonymous designs hide the real IDs of communicating parties during transactions. This is where proposed Pseudo trust design enters the picture.

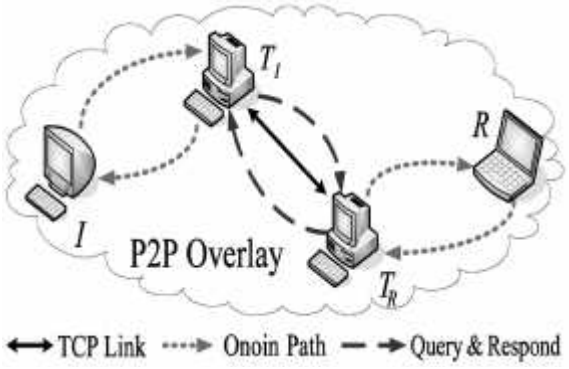

fig2 : PT query issuance and authentication

4.1 Design overview

PT is applicable under most of today's widely deployed P2P protocols, such as Gnutella and KaZaA. For simplicity of discussion, Gnutella-like P2P environment as a platform that PT runs on. In Gnutella, a requesting peer issues its queries in a flooding manner. A query is broadcast and rebroadcast until a certain criterion is satisfied. If the peer receiving the query can provide the requested object, a response message containing the IP address of the responder is sent back to the source peer along the reversed path of the query. To protect real identities, in the PT design, each peer is required to generate a *pseudo identity* (*PI*) before joining the system. As illustrated in Fig. 2, peers construct anonymous onion paths and find tail nodes based on the APFS protocol [1].

4.2 Pseudo identity generation and issuance

In PT, each peer is required to generate two items before joining the system: a *pseudo identity* (*PI*) and a *pseudo identity certificate* (*PIC*).A *PI* is used to identify and replace the real identity of a peer in a P2P system. In this way, a peer does not have to expose its real identity when communicating with others. Furthermore, a peer's reputation is also coupled
with its *PI* instead of its real ID. To avoid impersonation, *PIC* is generated to authenticate the *PI* holder.

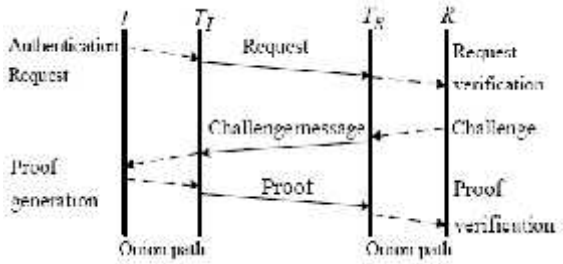

Fig3 :Procedure of zero knowledge proof

PI and PIC Generation: PT adopts a hash function, such as SHA-1, to generate *PI and PIC* for each peer. We slightly modify the prototype SHA-1 and name the revisions *hi* to fit the different inputs and outputs. A randomly chooses two large primes $p1$ and $p2$, and calculates the integer $n = p1 \times p2$. To protect the authenticity of *n*, we combine each *PI* with *n* through hash functions. [1].

4.3 New Peer Initialization:

After joining the P2P system, peer *A* constructs anonymous sessions with existing peers using the APFS protocol. In this design, anonymous sessions are onion routes (*OR*) consisting of chosen peers. A tail node *TA* acts as an agent relaying a message for *A*. *TA* and other peers in this path do not know *A* is at the end point position. *In APFS, peers use a multicast technique to send the query via a tail node to some servers anonymously, which is similar to the flooding procedure used by PT.* To allow *TA* to send messages back to *A*, *A* constructs an *OR*:{ ,{... ,{ } ,...} } *ORA = TA A mix KA KTA.* Meanwhile, peer *A* builds another onion path *A OR* for sending message to *TA* anonymously. After the construction of anonymous sessions (note each node selects its tail node and builds two onion paths), each peer can anonymously issue queries for desired files. We use *I* to denote an initiator. *I* forwards a query *q* for a certain requested file *f*, through *I OR* to its tail node *TI*. *TI* then starts a flooding search in the P2P system. When a peer receives the query and holds the requested file, it gets *I*'s credit based on the trust management mechanism to help it decide whether to act as a responder *R* and provide the file. If it decides to provide the file, *R* replies to this query through its tail node with a response including the IP address of its tail node and its reputation record.[2]

4.4 Authentication : PT employs a modified ZKP of Identification scheme. To adopt it into decentralized P2P networks, we remove the central authority servers in. ZKP used in PT is based on the assumption that factoring a large integer is computationally infeasible. When the query initiator, *I*, receives multiple responses, it selects those peers with high reputations as potential responders. Without loss of generality, suppose *R* is selected as one of the responders. Main steps in authentication procedure are described as follows:

1. Authentication Request
2. Request Verification
3. Challenge: Peer R sends a random binary vector to peer I.
4. Proof generation
5. Verification

Peer R accepts peer I's proof if the equality holds, otherwise it rejects the proof. After peer R verifies peer I, peer I verifies R.

4.5 Trust and Reputation Management

After completing the authentication procedure, peer I can download files from peer R. Similar to the authentication message exchange, the file can be delivered through anonymous sessions. A peer can either use the session key to encrypt the file or only encrypt the MAC of the file according to the users' security requirements. Since only peer I and peer R know their session keys, other peers, even the tail nodes, cannot forge the file to deceive the initiator during the data transmissions. After

downloading a file, peer I can evaluate the file and provide comments to peers who provide resources such that most existing trust management mechanisms, such as EigenTrust [4] and XenoTrust [4], become applicable. The only difference after employing PT in these systems is that the reputation of peers will be connected with peer pseudo identities (PIs) instead of their real IDs or IP addresses.[4].

4.6 Pseudo Trust in Structured Peer-to-Peer System

PT can also be adopted in structured P2P systems. In structured P2Ps, such as CAN and Chord, each peer's IP address and resource can be mapped to a point in the ID space using distributed hashing tables (DHTs). Each peer stores a fraction of the entire ID space, for example a zone in CAN. Lookup processes are handled as follows: On the one hand, the resource index is mapped to a point in the ID space via DHT by the resource holder, for example the Insert(key, value) process in CAN. The index, with the IP address of the resource holder, will be maintained by the peer whose zone covers the point. On the other hand, a requester computes the key of desired resource and issues it to DHT, for example the Lookup(key) process in Chord, to obtain the index of desired resource from the peer holding this information. Then, the requester will directly contact the provider to retrieve the resource. In this procedure, there is no anonymity protection for peers PT can be applied in structured P2P systems. First, the core technique of structured P2P systems is the consistent hashing function, which has features including the uniform distribution of outputs and resilience to collisions. These features are also essential properties of cryptographic hash functions1 [4], which have been employed in the PT for PI generation. The typical consistent hash function used in structured P2P systems is MD-5 or SHA-1. In PT, the PIs of users are also generated by using SHA-1 or MD-5, which enables PIs of PT to be used in structured P2P systems with slight modification. In addition, adopting PT in structured P2P system can enhance the anonymity of peers. Structured P2Ps such as CAN or Chord generate the peer's identity through DHTs but are "limited" to achieve anonymity. In most structured P2Ps, a peer's ID is derived by hashing the peer's IP address. Peers do not input any unique secret into the hash function to generate the IDs. Therefore, any peer who knows a given IP address can easily generate the corresponding ID. This process makes peers in CAN or Chord suffer from impersonation. It thereby degrades the anonymity and security and cannot guarantee the validation of authentication. In PT, peers use their unique secrets as an input of ID generation. It makes the pseudo ID verifiable and invulnerable to impersonation. By using PT authentication mechanism, the validation of a pseudo ID is guaranteed. Thus, replacing the original ID with a PI via PT protocol will enhance the anonymity and the security in structured P2P systems. Second, instead of embedding an IP address in the resource index, the resource provider can pre construct an onion path, which points to itself, and attach it to the resource index.

5 SECURITY ANALYSES

This section analyze of the reply attack, MIMA and Denial of Service Attack.

5.1 Replay Attack

For a replay attack, malicious peers collect some previous proofs of an initiator and resend these proofs to the responder. To convince the responder, malicious peers must guess the challenge message e generated by the responder completely and correctly.According to the proof of Lemma 2, the probability of a malicious peer's guessing correctly a challenge message e chosen by the responder is 2_k. Thus, the success probability of a replay attack is 2_k.

5.2 Man-In-Middle-Attack

A MIMA is an attack in which an intruder M is able to arbitrarily access and modify messages between two parties without either party knowing that the link between them has been compromised. As a result, M can successfully impersonate the initiator to the responder, or vice versa. To PT users, intruders can modify and relay the forged authentication messages to participants and try to convince peer I or peer R that M is the opposing party.

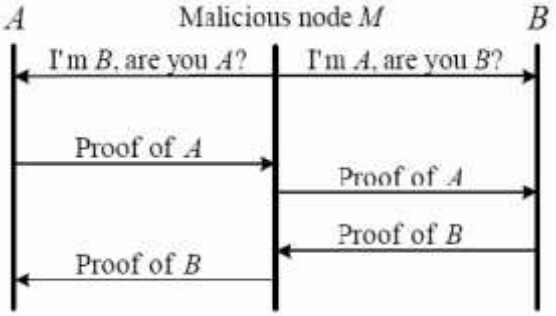

Fig4. MIMA attack. *M* cheats *A* or *B* that *M* is the real opposing party.

5.3 Denial of Service Attack

DoS attack has gained increasing concern in distributed networks. Typically, this attack renders networks, hosts, and other victim systems unusable by consuming the bandwidth of victim networks or deluging them with a huge number of requests to overload their systems. In the design of PT, the system may suffer from DoS attacks during authentication interactions. Since *I* chooses a desired responder from the returned responses to start an authentication procedure, I does not suffer from DoS attacks. Thus,simply focus on the possibility that responders are under DoS attacks. Indeed, the authentication sequence of PT is well designed to defend against DoS attacks. This kind of DoS attack may devour the computational resources or available network connections of R and TR and render them unable to answer legitimate requests until the attack ends. To avoid this possible drawback, PT orders that I must prove itself to R first, and then, R conducts the verification of I. Therefore, if attackers conduct DoS attacks on R by propagating numerous forged authentication requests, they must generate the same amount of valid PIs in advance. Otherwise, forged PIs would fail to pass the verification of R. For each verification request, R does nothing but generate a random number e for each I before it completes the proving step. That is, the attackers should not only try to generate enough valid PIs but also finish the proving phase on their side.

Since the computational overhead in the verification process is relatively small compared to that in the proving procedure, a DoS attack hardly has an effective influence on R. Moreover, valid PIs used to perform DoS attacks would degrade the reputations of those pseudo identities. To continue DoS attacks, adversaries have to generate more valid PIs. Even if adversaries form a clique to boost each other to corrupt the

reputation system, the overhead of DoS attacks is quite expensive and unacceptable. Thus, the design of PT effectively defends against DoS attacks.[4].

6. PERFORMANCE EVALUATIONS

Evaluate the PT design by trace-driven simulations, in which the P2P topologies are obtained from the DSS Clip2 trace. Here simulations are performed on those traces in a variety of network sizes ranging from hundreds to thousands. For each simulation, take the average result from 1,000 runs. The results are consistent with traces of different days, and here, we show the representative results. P2P networks are highly dynamic, with peers frequently joining and leaving. Now simulate the joining

and leaving behaviour of peers by turning on and off logical peers, respectively. In our simulation, each node issues 0.3 queries/min. When a peer joins, a lifetime in seconds is assigned to the peer. The lifetime of a peer is defined as the time period that a peer stays in the system. The lifetime is generated according to the distribution observed in [4]. The mean of the distribution is chosen to be 10 minutes.

6.1 Response Time

Of all latencies in a P2P system, the response time from query issuance to the start of the download is of greatest concern, as it has a significant bearing on the system usability.[1]

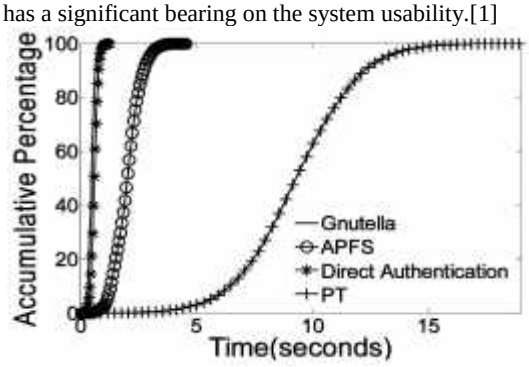

Fig 5.Response time

Fig. 5 plots the simulation results of the response time, which shows the accumulative percentage of returned responses versus time for PT, overt Gnutella protocol, and APFS.

The comparison in Fig. 5 shows that the response time of APFS is approximately three times that of overt Gnutella, while PT is around seven times that of overt Gnutella. In APFS, users need one onion path plus a flooding procedure to send a query out, one TCP link to deliver the response between tail nodes, and two onion paths to send the response anonymously. In PT's two-phase authentication procedures between two parties, the numbers of used onion paths and TCP links are 12 and 6 to follow APFS, respectively. According to the design of PT, the authentication messages pass through the TCP connections between two tail nodes six times in a mutual authentication procedure. Our observation shows that the average response time of normal query flooding, direct authentication, APFS, and PT are about 493, 600, 2,031, and 9,296 ms, respectively. Note that the time consumed in anonymous paths of PT constitutes a major part of the whole latency, which is four times more than that of APFS. Therefore, the time consumption of authentication is indeed trivial.

6.2 Scalability

As the size of the P2P overlay grows, PT has good scalability in the response time and the traffic overhead. This is due to the fact that 1) no central server or CA is involved during our authentication procedures, 2) the design is efficient and the extra computation needed is

Relatively trivial and 3) the underlying anonymous protocol we have selected, APFS, has good scalability. Now increase the size of the P2P overlay from 500 to 13,000 and observe the average response time of 10,000 queries. Results show that the growth of the P2P overlay has little impact on the PT performance, as shown in Fig. 6. [1]

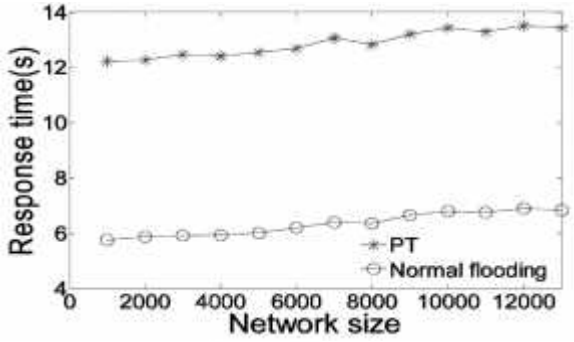

Fig 6: Scalability of PT**.**

In fact, PT employs an authentication procedure through the direct TCP connection and the delay is mainly related to the average distance between two nodes in the physical Internet layer.

7. PROTOTYPE DESIGN AND IMPLEMENTATION

The first set focuses on the extra computation overhead caused by PT and tests the computing capabilities of normal PCs running this protocol. The second set tests the overall latency of pseudo identity authentication procedures in the Internet environment.[1]

7.1 Overview

Let's develop PT prototype programs over VC6 and Win XP SP2 platforms. To shorten the development cycle, by using Victor Shoup's NTL [4], a number theory library, this has gained wide acceptance in the cryptography community for large integer operation. The prototype P2P servant modifies Gnutella 0.6 by adding three major components: System initialization, Prover , and Verifier toolkit. The System initialization deals with new peers joining, including the PI and PIC generation and issuance, parameters setup, anonymous path construction, and so on. The main authentication algorithms and their operations are conducted by the Prover and Verifier toolkit. To generate high-quality random and nondeterministic numbers, PT allows users to grab the system clock and mouse movements as the random seed resource.[4]

7.2 Design approach

To authenticate a user, ZKP protocols require more than two steps: a user's request, a server's challenge, a user's response, and a server's response. Due to the ZKP nature, the server's challenges cannot be delivered to the user with the login form. Therefore, such protocols differ from the classical approaches, and require more flexible communication means. In addition, the authentication process is more computationally expensive and consumes more bandwidth. To satisfy these requirements, we decided to combine Ajax (Asynchronous JavaScript and XML)4 that is an asynchronous web technology with a graph isomorphism protocol. We chose this NP problem since it does not require user's browsers to find co prime numbers nor multiplicative inverses. Computations are performed using natural numbers and matrices; therefore, such arithmetic is easier to implement than, for example, elliptic curves. This technology choice did not require plug-in and was supported by default settings of the vast majority of web browsers. Hence, integration with existing applications was straightforward. Furthermore, their start-up time was negligible. [5]

7.2.1 Authentication procedure

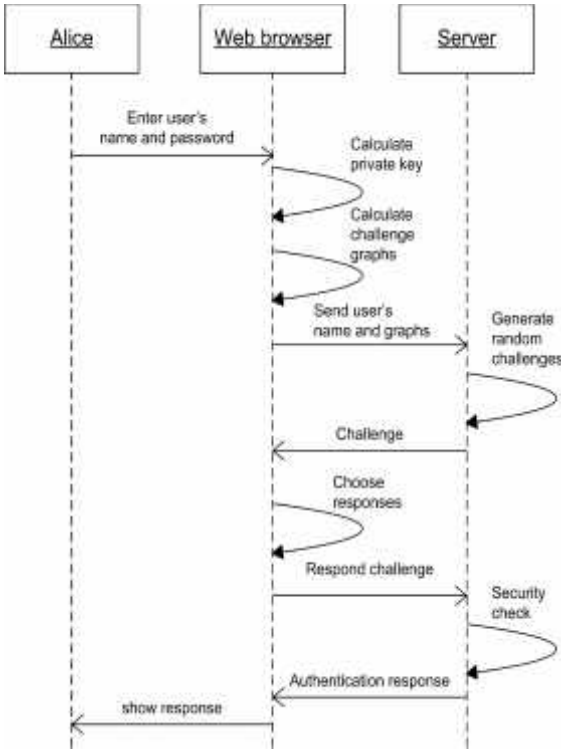

Fig 7. ZKP Implementation on the Web

A sequence diagram presented on Figure 7 shows our approach in detail. Alice types her username and password, but the password never leaves her browser. In contrary to existing approaches like HTTP MD5 digest, the server does not have any information that would allow for impersonation Alice at another server. The browser uses the password to calculate her public-private key pairs and then executes the ZKP protocol. The browser is responsible for a number of new tasks: calculating private keys from passwords, generating challenge graphs, and responding to the challenge. A server has only one more responsibility: generating random challenges. There is also one more interaction between a browser and a server in comparison with classical approaches Thus, the main question is if the new approach is feasible and will not require long waiting times for users.

7.3 Implementation in Internet Environments

To do PT prototype in an overlay comprising of 50 desktop PCs at the laboratories of CAS,the campus network of HKUST, and other sites. The representative configuration of each machine includes a PIV1.8G CPU, 256 MBytes of memory, and a 1000M Ethernet card. To better evaluate PT, in this implementation, ignore the time consumed by the APFS protocol.[1]

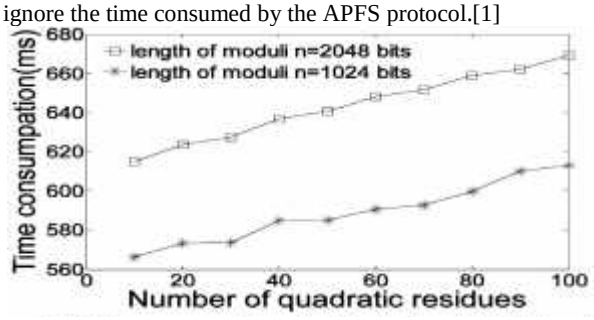

Fig8: Time consumption of authentication in the Internet

To easily adopt PT in current P2P systems, here included some reliable modulus in API. The primary version of the PT package is available on our public Website [4].Fig. 8 shows the experimental results in the Internet. The number of quadratic residues ranges from 10 to 100, and we use 1,024 and 2,048 bits as moduli sizes, respectively. Employ ping tests with packets of the same size as PT messages over the two involved computers between Hong Kong and Beijing in the Internet. The results range from 0.07 to 0.12 second. Therefore, the overall latency of PT is less than 0.67 second, which is relatively small for an authentication procedure. Note that the extra latency of PT in implementation results is much shorter than that in the simulation results because the time consumed by the Gnutella and APFS protocols is included in the simulation but not in the prototype implementation.[1]

8 CONCLUSIONS

Due to the inherent trade-off between trust and anonymity, existing attractive identity-based trust management schemes cannot be directly employed in anonymous P2P systems. Here propose an anonymous zero-knowledge authentication protocol, called Pseudo Trust. In this design, a ZKP-based authentication scheme is designed to support trust management in anonymous P2P systems, so that peers may use un forgeable and verifiable pseudonyms instead of their real identities in P2P communities. We prove that the probability of a successful impersonation is computationally infeasible, even if the adversaries have collected all of the previous authentication messages. This manages to address man in- middle-attacks in the PT design. The results of trace driven simulations show that PT has perfect scalability in both static and dynamic environments. Here implemented prototype of PT and evaluate its performance by experiments. The wide deployment of Pseudo Trust will provide better privacy and security for P2P users.

By balancing the security needs, applications and system capabilities it may still be possible to build systems with relatively good security, instead of no security as in most current systems. It is evident from this study, that implementing real security does have quite large computational and memory requirements. So, in applications where good security is necessary, high-powered embedded controllers should be selected, so that they can work with the full-strength cryptographic protocols. Another possible solution would be to use symmetric-key cryptography protocols (which have relatively small computational and memory requirements), with designed-in features for the eventual loss of key secrecy through reverse engineering. Joining theoretically good cryptographic techniques and protocols with real world limitations of small systems is sometimes extremely hard. Most studies are based on the availability of enough computing power and memory. When you try to apply these techniques in very small systems, you will have to make compromises. Knowing the strength of your protocols, keys and the hardware and being able to balance and apply the system to your actual needs will be even more important than in traditional cryptographic applications.

REFERENCES

[1] Li Lu, Jinsong Han, Lei Hu, Jinpeng Huai, Yunhao Liu, and Lionel M. Ni , Pseudo Trust: Zero-Knowledge Based Authentication in Anonymous Peer-to-Peer Protocols , *in*
IEEE transactions on parallel and distributed systems,2007.

[2] Cao Lai-Cheng, Heightening Security of P2P Networks by Neighborhood Key Method, First International Conference on Intelligent Networks and Intelligent Systems,2008

[3] Abdullah M. Jaafar, Azman Samsudin, Visual Zero-Knowledge Proof of Identity Scheme: A New Approach , Second International Conference on Computer Research and Development,2010

[4] Li Lu, Jinsong Han, Lei Hu, Jinpeng Huai, Yunhao Liu, and Lionel M. Ni , Pseudo Trust: Zero-Knowledge Based Authentication in Anonymous Peer-to-Peer Protocols , *in*
IEEE transactions on parallel and distributed systems,2008

[5]. Sławomir Grzonkowski, Wojciech Zaremba, Maciej Zaremba, Bill McDaniel, Extending Web Applications with a Lightweight Zero Knowledge Proof Authentication, CSTST , France ,2008,

[6] D. Qiu and R. Srikant, "Modeling and Performance Analysis of BitTorrent-Like Peer-to-Peer Networks," Proc. ACM SIGCOMM, 2004.

[7] Mohammad Ahmad Alia and Azman Bin Samsudin, Fractal (Mandelbrot and Julia) Zero-Knowledge Proof of Identity, Journal of Computer Science 4 (5): 408-414, 2008
[8] Hannu A. Aronsson, Zero Knowledge Protocols and Small Systems, Network Security: Zero Knowledge and Small Systems,http://www.tml.tkk.fi/Opinnot/Tik-110.501/1995/zeroknowledge.html, 1995
[9] jean –jacqus,myriam,Michael,Muriel,Louis ,marie ,anna,gaid,gwenle,soazig,tom berson,;how to explain zero-knowledge protocols to your children,CCETT/EPT France,1998
[10] J. Brandt, I.B. Damgard, P. Landrock, and T. Pedersen, "Zero-Knowledge Authentication Scheme with Secret Key Exchange," Proc. Advances in Cryptology (CRYPTO), 1990.
[11] *Annarita Giani*, Security Essentials - Version 1.2e, IDENTIFICATION WITH ZERO KNOWLEDGE PROTOCOLS, SANS Institute InfoSec Reading Room,2001
[12] Gerardo I. Simari, *A Primer on Zero Knowledge Protocols*, Universidad Nacional del Sur - , 2002
[13] Jos_e Bacelar Almeida, Endre Bangerter, Manuel Barbosa, Stephan Krenn, Ahmad-Reza Sadeghi, Thomas Schneider, A Certifying Compiler for Zero-Knowledge Proofs of Knowledge Based on E-Protocols, European Community's Seventh Framework Programme (FP7) under grant agreement no. 216499. 2006
[14] P. P. C. Lee, J. C. S. Lui, and D. K. Y. Yau. Distributed Collaborative Key Agreement and Authentication Protocols for Dynamic Peer Groups. *IEEE/ACM Transactions on Networking*, 2006.
[15] G. Jain, "Zero knowledge proofs: A survey. Tech. reports," University of Pennsylvania, 2008.
[16] Juels, A., Pappu, R., *Squealing Euros: Privacy Protection in RFID-Enabled Banknotes*, Seventh International Financial Cryptography Conference, Gosier, Guadeloupe, January 2003
[17] A. Narayanan and V. Shmatikov. Fast dictionary attacks on passwords using time-space trade. InCCS '05: Proceedings of the 12th ACM conference on Computer and communications security, pages 364{372, New York, NY, USA,} 2005.
[18] V. Scarlata, B. N. Levine, and C. Shields. Responder Anonymity and Anonymous Peer-to-Peer File Sharing. In Proceedings of *IEEE ICNP*, 2001.