A Device Authentication Mechanism Reducing Performance Cost in Mobile P2P Networks

Yoon-Su Jeong[1], Yong-Tae Kim[2*], Seung-Soo Shin[3] and Sang-Ho Lee[4]

[1] Department of Information Communication Engineering, Mokwon University
88 Doanbuk-ro, Seo-gu, Daejeon, 302-729 - KOR
[e-mail: bukmunro@mokwon.ac.kr]
[2] Department of Multimedia Engineering, Hannam University
133 Ojeong-dong, Daeduk-gu, Daejeon - KOR
[e-mail: ky7762@hnu.ac.kr]
[3] Department of Information Security, Tongmyong University
428, Sinseonno, Nan-gu, Busan, 608-711 - KOR
[e-mail: shinss@tu.ac.kr]
[4] Department of Software Engineering, Chungbuk National University
52 Naesudong-ro, Heungdeok-gu, Cheongju Chungbuk 361-763 - KOR
[e-mail: shlee@chungbuk.ac.kr]
* Corresponding author : Yong-Tae Kim

Received March 28, 2012; revised June 26, 2012; revised February 4, 2013; revised March 27, 2013; accepted April 8, 2013; published April 30, 2013

Abstract

The main concern in mobile peer-to-peer (P2P) networks is security because jamming or eavesdropping on a wireless link is much easier than on a wired one and such damage can be incurred without physical access or contact. In particular, authentication has increasingly become a requirement in mobile P2P environments. This paper presents a new mutual authentication mechanism which requires less storage space and maintains a high level of security in mobile P2P networks. The proposed mechanism improves efficiency by avoiding the use of centralized entities and is designed to be agile in terms of both reliability and low-cost implementation. The mechanism suggested in the simulation evaluates the function costs occurring in authentication between the devices under mobile P2P network environment comparing to existing method in terms of basic operation costs, traffic costs, communications costs, storage costs and scalability. The simulation results show that the proposed mechanism provides high authentication with low cryptography processing overhead.

Keywords: mobile Peer-to-Peer (P2P), mutual authentication, security

http://dx.doi.org/10.3837/tiis.2013.04.019

1. Introduction

Peer-to-Peer (P2P) is a generic term assigned to network architectures in which all nodes offer the same services and behave in an identical manner [24]. The advantage of P2P systems [23, 25] is scalability in supporting millions of nodes easily, cooperatively storing and replicating data using distributed hash tables**,** and allowing messages to be routed efficiently among network nodes. These underlying techniques enable arrays of applications to be growing and exciting, including distributed data storage, distributed bandwidth sharing, and multicast data distribution.

The rapid growth of mobile devices with diverse functions has created mobile P2P systems with limited computational and battery resources. Recent advancements with wireless networking and mobile computing technologies, such as wireless LANs, wireless mesh networks, and 3G cellular networks, have further facilitated the migration of the P2P paradigm into wireless mobile computing [25]. The combination of mobile technologies with P2P is an ideal technology for organizations with characteristics such as a decentralized management style, or having highly mobile workforces who are geographically dispersed requiring a wide range of computing and communications devices.

The ad-hoc and heterogeneous nature of mobile P2P systems, however, can present significant challenges to application designers in charge of security and privacy [9]. Within a mobile P2P system, encryption must be used. Robust authentication procedures are also required to be connected to trusted devices with non-trusted ones. A task will be difficult in decentralized environments, where connection to a trusted authority is not guaranteed. Theoretically, no node in a mobile P2P environment has any knowledge of peers two or more hops away. Based on this observation, a mobile P2P network is able to achieve partial anonymity, at least among those non-neighboring peers. However, current mobile P2P protocols fail to provide real anonymity guarantees. As queries are received in plain text, the contents of these messages are exposed to malicious nodes, and attackers can easily guess the identities of the communicating parties [8].

The peer-to-peer Personal Privacy Protocol (P5) [7], based on a global broadcast channel, aims to achieve mutual anonymity. Mutual anonymity is defined as a situation in which an initiator sends a request for a service without knowing which node actually provides the service. Likewise, the responder sends responses without knowing the identity of the initiator. While P5 assumes that the initiator knows the public key of the query responder, it is not easy to be used in practice. Some tail nodes of Anonymous Peer-to-peer File Sharing (APFS) [10] are designed for mutual anonymous communications that act as anonymous proxies. Then, initiator peers anonymously contact servers to send requests and receive data through tail nodes and onion paths. Xiao et al. [9] provides an anonymity solution using a shortcut responding protocol in pure P2P systems. In this protocol, an initiator binds an onion-structured return path with each query. Each peer that receives the query probabilistically decides whether or not it will act as the query agent node. The advantages of this work include shorter-than-normal return path patterns and a high degree of security for the RSA-based encryption method.

Mutual authentication assumes that two devices agree with each other regarding the value of a public data string D. The data string D can be the concatenation of the public keys of A and B for an asymmetric cryptosystem. It can support the registration process for a small-scale PKI, or can simply be used as the basis for subsequent secure communications. In order to ensure mutual authentication between mobile devices, a new device authentication mechanism

for mutual authentication guarantees secure communication between arbitrary mobile devices is needed. The main goal of this study is to achieve anonymous authentication between mobile devices without the use of centralized entities.

This paper reviwes significant studies related to device authentication and mutual authentication in P2P systems. Then, a proposed device authentication mechanism will be introduced for mutual anonymity in overlay networks. Performance evaluations are presented on the costs of the proposed device authentication mechanism. Finally, the security efficiency of the proposed approach will be discussed along with conclusions of the study.

2. Related Work

This study focuses on mutual anonymity between two devices. We would like to review recent papers related to mutual anonymity in overlay networks in this section.

2.1 Characteristic of Device Authentication

Device authentication provides many benefits and enhances network security at a very favorable return on investment [19] and by adding another layer of protection to the defense-in-depth strategy. Device authentication allows only authorized users, having previously enrolled devices, to enter a network and access data. The authentication procedure permits organizations to synchronize their user and device policies. Furthermore, device authentication integrates the secure identification of authorized desktops, laptops, and other remote entry devices into a comprehensive organizational security strategy at a very reasonable cost [20, 21]. The authentication is effective in securing remote access by mobile users and home office users who must access the network through a Virtual Personal Network virtual private network (VPN) or other remote connection utilizing a High Assurance Remote network. Finally, the device authentication can be a strategic and enabled technology for e-governments and other agency applications because device control, in conjunction with user control, is the main issue regarding security in these fields.

P2P device authentication systems such as Gnutella, KaZaA, and BitTorrent, employ a routed-search-and-direct-download mechanism [18, 32]. Client-server authentication using a traditional public-key may be open in the sense that any users, even strangers, can authenticate the server identity using its public key certificate. In contrast, P2P device communication is often a closed system. Generally, peer devices must have a credentials setup first, often out-of-band by an administrator, before they can join the communicating group.

Credentials setup indicates the initial stage for peers to receive verification. The credentials are verified through a password, personal identification number (PIN), smart card, one-time password generator, software or device containing secret key or personal key, finger print or retina recognition. In such a symmetric and closed system, a pre-shared key (PSK) authentication method is commonly used for its simplicity. The weakness of conventional PSK methods is that a common secret is shared among multiple entities. These methods distribute multiple shared secrets. This creates a key distribution challenge up to N because N-1 shared keys are needed for N peers.

2.2 Mutual Authentication Protocols

Many studies have focused on P2P systems [11-17]. However, two recent approaches have provided a publisher anonymity protocol [2,6]. The first approach employs a hash to mark the key information of documents as Freenet [2]. As used in Publius [6], Freehaven [3,5], GNUnet

[4], and Gap [1], the second approach uses a scheme similar to Shamir secret sharing to either split a symmetric key or break the file into n shares to achieve the goal of anonymous file sharing.

Sherwood et al. [7] first proposed the use of a global broadcast channel to achieve mutual anonymity, in which all participants in an anonymous communication exchange send fixed-length packets onto this channel at a fixed rate. This protocol pays special attention to eliminating the possibility of determining the communication linkability between two specific peer nodes by providing equal and regular broadcast activities among the entire peer group. The broadcast nature of this framework can limit the size of the communication group. To address this limit, the authors further proposed the P^5 scheme, which creates a hierarchy of broadcast channels to make the system scalable. The basic idea of P^5 is to permit all participants in the channel to send fixed-length encrypted packets at a fixed rate as if all participants were in a logic ring. However, P^5 does not provide the high bandwidth efficiency outlined in [8].

APFS [10] is designed for a decentralized system such as Gnutella. APFS allows new peers to join and leave the system periodically by sending a message to a coordinator. Some coordinator nodes act as a superior peer and maintain a list of all peer nodes. The coordinator responds with a list of current servers. Some peers in these lists volunteer to issue queries for others. Then the initiator sends a match request to a path in which the tail node is the last member. There are two advantages of using APFS. First, all the communications in the system are mutually anonymous. Additionally, the anonymous protocols are designed for a pure P2P system in which trusted centralized servers may not be available. However, APFS has some disadvantages: first, the suitability of a volunteer peer needs to be taken into account, since this factor can significantly affect the performance of P2P systems. Second, the number of servers can dynamically change. Third, since a trusted server cannot be guaranteed, anonymous communications can become highly complicated.

In the shortcut-responding protocol [9], the initiator establishes an onion-based reply block known as a re-mailer before sending a query. A re-mailer is like an anonymous return path. Each peer that receives the query determines whether it will agree to be a query agent peer with the probability of PV. If a peer acts as the query agent for the initiator, it floods this query into P2P systems. Upon receiving a request, a responder builds another onion path to send the file to the query agent peer anonymously. The query agent peer delivers the file along the return path to the initiator. Although reducing the length of the return path, this approach does not consider the reply-confirm procedure between the initiator and the responder. Another reason it cannot be directly utlized in P2P networks is because of the assumption that every initiator knows the public keys of all possible responders.

No scheme proposed thus far supports both mutual authentication and anonymity between a dynamically changed server and a device with mobility. Mobile devices can apply to various environments, depending on the function of the device, without limits to the area in its operation. A scheme that supports only a limited communication group using a hash does not support all of the anonymous needs of devices that are frequently mobile. When the central server dynamically changes, unlike with APFS, it does not guarantee anonymous communications. To respond, it guarantees the anonymity of the device by applying a hybrid elliptic curve cryptography (ECC) algorithm that communications between devices after dividing the entire network into several cluster groups and also registering the device information once per device for devices that play the role of the cluster head.

3. Device Authentication Mechanism for Mutual Anonymity

In this section, a new Device Authentication Mechanism for Mutual Anonymity (DAMMA) using an elliptic curve (EC) is proposed because DAMMA guarantees secure communication between arbitrary devices and offers an authentication method for an overlay P2P system. In order to perform authentications between devices smoothly using DAMMA in a heterogeneous environment, the necessary condition is to register devices to a certification authority(CA) that already has the authority of authentication. The registered devices, through a node beacon that is sent by the CA, inform the CA whether they will participate in the network that has a cluster. Through this, the CA identifies the devices participating in the network. The CA then distributes a certificate to the devices identified as participating in the network in an effort to guarantee the integrity of the devices. To send and receive information with other devices, the devices participating in the network establishes a security association (SA). This ensures safe communication between the devices using a key known by the server whenever communication is required between devices in heterogeneous environments. A heterogeneous system, each with its own requirements, usually generates the different properties of device data and can be integrated without difficulty and violating other systems' missions. Moreover, whenever communication is required again by devices after earlier communication has terminated, verification of the device integrity is completed and this determines whether information on the devices registered on the server existed initially. Communication is then resumed to establish a SA. The required assumption is that the procedures required for DAMMA are equivalent to those for distributed management schemes. Additionally, the devices are owned by users, and biometric information of these users is stored.

3.1 Anonymous Communication Access

We assume that DAMMA organized into a virtual overlay network, forms a unique true group, in the sense that preliminary trust relationships have been established. DAMMA divides all groups into several groups to increase the connecting ability of each key and the maximum supportable network size. Particularly, special nodes (SNs), called super nodes, are selected to act as authentication servers. These servers are responsible for security establishments within a limited cluster of nodes, as shown in Fig. 1. The peer that desires to store a file is equipped with a zero-knowledge value. After the storage operation, this value will enable only the correct peer to modify the previously stored file. Using the DAMMA protocol, the authentication information cannot be used by a peer that routes the message for its own purpose.

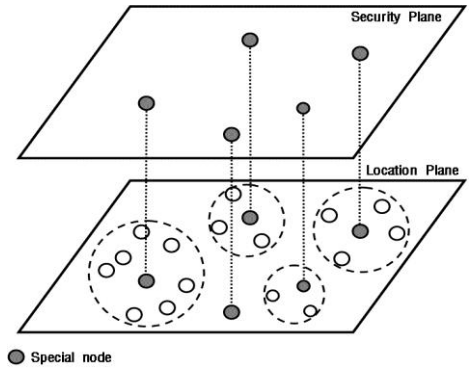


Fig. 1. An overview of the proposed scheme

In a P2P network, multicasting can be implemented in various ways. Fig. 2 shows the proposed P2P multicast model. As shown in Fig. 2, the proposed P2P system implements an overlay network (i.e., an extra network layer for multicast supports called the P2P multicast network layer) on top of the native P2P network layer. Forwarding multicasting is carried out via a multi-hop unicast or multi-destination unicast. Therefore, only a few nodes in the peer-to-peer network need to support the multicast.

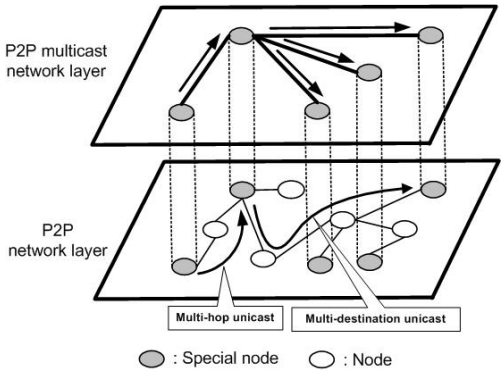


Fig. 2. P2P multicast communication

Fig. 3 depicts how multicasting works in the proposed P2P model. When a P2P node wants to join a particular multicast group, the steps carried out are 1) the joining P2P node sends a "join" message to the nearest node that is a member of the concerned multicast group; 2) the nearest member node returns a response message to acknowledge the joining; 3) a logical link for multicasting is formed between the two P2P nodes; and 4) multicast packets are propagated along to form logical links, each of which corresponds to a multicast path. The nearest multicast group member is found by flooding search requests out across nodes in the overlay (nodes connected by logical links). In the physical network layer, the sequence number of multicast nodes are used to inform the nearest multicast member node that responds to the query of joining nodes.

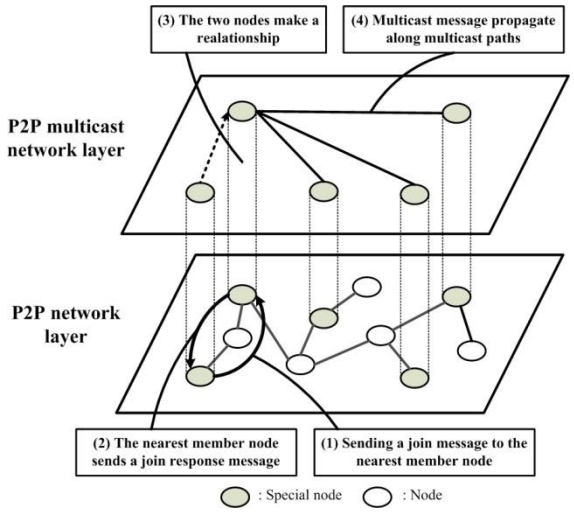


Fig. 3. Procedure of joining a multicast group

Each peer node in the proposed P2P network model maintains information about other P2P nodes and their users. Peers join a certain multicast group to share their resources with other peers participating in that group, and messages intended for the multicast group are exchanged via a P2P multicast or a unicast transmission. The proposed P2P model can take advantage of the failure of recovery mechanisms with regard to dynamic multicast communications (i.e., joining/leaving a multicast group is flexible). Thus, it is more resilient to P2P node failures. Fig. 4 shows an example of a P2P multicast application.

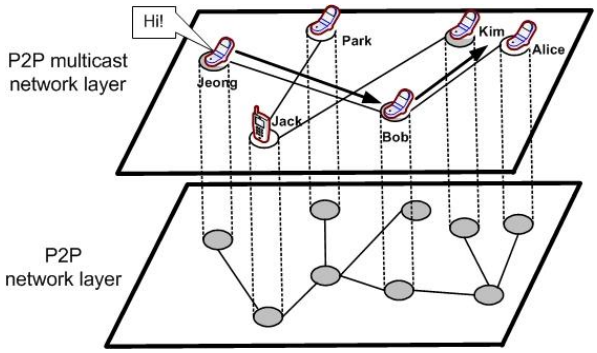


Fig. 4. P2P instant message application

3.1.1 Local Anonymous Access

When an arbitrary device requests a local anonymous access in a specific local region of a P2P environment, the device first checks if there is a device with a resource in the same autonomous system (AS) region. If there is a device in the AS region, the arbitrary device uses the open key Q_x to verify anonymous identity (AID) after recognizing the signature SN node in the AS. The open key Q_x has the same function as identity (ID) in the proposed protocol. Once the arbitrary device succeeds in verifying the AID, the device with the resource carries out the anonymous multicast to send the anonymous resource to the arbitrary device.

3.1.2 Multi-domain Anonymous Access

If device V1s in the local AS have no information for the resource requested by arbitrary device U, they release the query of device U by using the same protocol as SSMP [29], PUZZLE[33], NMA[34], RR[35]. For the opponent V2 in the other AS region to recognize the signature of other SNs, an additional mechanism must be secured. If device V1s successfully receive the information of other local ASs selected by the SN, they save the SN recognition information in a table called the 'SN-table'. In the SN-table, the SN information of other AS regions is recorded again. The opponent normal device V2 with the resource receives the query and compares it with the local SN-table. If the query information is identical with that in the SN-table, the opponent device V2 verifies the signature of arbitrary device U. If the opponent device V2 finds the SN signature and the message information for time stamp T, the opponent device V2 passes the verification. However, the attacker may accept the opponent device V2 in disguise so that the device V1 does not operate normally. After all, to contact the device U anonymously, the opponent device V2 uses a protocol such as SSMP, PUZZLE, NMA, RR and sends the resource to the device U. However, the information reception probability depends on the transmission distance. In some cases, the opponent devices may keep forwarding the verification until the information in the SN meets the device with the resource. Of course, the opponent device applies the previous signature verification to validate the normality of the device before it relies on the device.

3.2 Notation

The notations used in this paper are summarized in Table 1. Let q denote the order of the underlying finite field F_q and let E be a suitably chosen elliptic curve defined over F_q. Let P denote a base point in E, the generator point, and n be the order of P, where n is prime. Thus, a point P is a point satisfying nP =0, where 0 is the point at infinity. It is assumed that the discrete logarithm problem in the group $< P >$ of points generated by P is intractable. Let q_{CA} $\in$ [2, n -2] be a random integer selected by the certification authority (CA) and Q_{CA} = $q_{CA} \times P$. The pair of the static secret/public key pair of the CA is q_{CA}, Q_{CA}.

The CA generates a network-wide symmetric key K, which will be used by all nodes as an initial authenticator in order to avoid processing counterfeit "hello" messages and to prevent trivial DoS attacks. Furthermore, the CA also generates a set of independent symmetric encryption keys, K_1, K_2, ... , K_m, one key for each of the m node generations. These keys are similar to the generation keys of the LEAP protocol [27]. However in the proposed mechanism, these keys are only used to create a temporary channel for exchanging randomness for key establishment procedures, to mitigate the consequences of a static key being compromised and also to establish forward secrecy (privacy) for exchanged session keys. The description of the DAMMA protocol is as follows:

1. It is assumed that network links are bidirectional, i.e., if node A can accept node B, B can also accept A. This is true when all the nodes use omnidirectional antennas and have equal power levels.

2. It is not assumed that a central key server exists in the formed network, whereas it may exist off-line to initiate the nodes prior to the formation of the network.

Table 1.. Notations

Notation	Description
A, B	Two generic device nodes in P2P networks
AID_x	Anonymous identity of entity x
DI_x	Device identity information of entity x
E_1 , $E_1^{'}$, E_2 , $E_2^{'}$, Q , $\overline{Q}$	Middle values of key distribution process
T_i	Timestamp denotes the current time
E	Curve over $GF(q)$
P	Base point in E
q	Order of F_q
q_x	Temporary random secret key of entity x $q_x \in \{2, \ldots, n\text{-}1\}$
Q_x	Temporary public key of entity x $d_x \cdot P = Q_x$
$< P >$	The subgroup of E generated by P
M_x	Message generated by entity x
SK	Session key
CA	Certificate authority
SA	Security Association
AK_x	Authentication key of entity x
r_x	Random key of entity x
PU_x	Public key of entity x
PR_x	Secret key of entity x
PE_x	Encryption using public key of entity x
PD_y	Decryption using secret key of entity y
ID_x	Identity generated by entity x
E_K	Encryption using symmetry key K
$h()$	Hash function
$H_i(\cdot)$	Secure one-way hash functions
$H_1(\cdot)$	One-way hash function : $\{0,1\} \rightarrow G_P$
$H_2(\cdot)$	One-way hash function : $\{0,1\} \rightarrow Z_P^*$
$\|\|$	Concatenation
$\oplus$	XOR operation

3.3 DAMMA Protocol

The DAMMA protocol was designed to be cluster-based for use in a situation in which one device (A) communicates with another (B) in a heterogeneous environment. It is also assumed that the two devices will agree on the value of the public data string D. This data string can be a concatenation of the public keys of A and B for an asymmetric cryptosystem. This can support the registration process for a small-scale PKI or may simply be used as the basis for subsequent secure communications. To adopt PKI, anonymous systems remove the CA. Devices simply use their public/secret keys as pseudonyms. In particular, the public keys can be used to provide the basis for an authenticated secret key establishment protocol, requiring no further intervention by the mobile devices. DAMMA consists of two processes: one is a registration and cluster formation process and the other is a device authentication protocol

process. Fig. 5 shows the major operations in which device A communicates with device B in a heterogeneous environment.

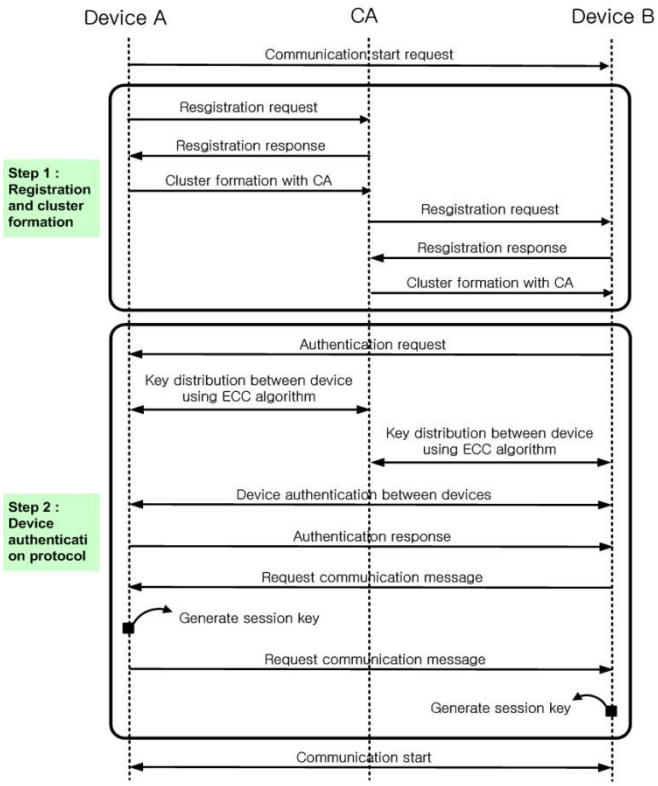


Fig. 5. Overall authentication operation in DAMMA

3.3.1 Registration and Cluster Formation

To build environments to perform the authentication protocol among devices in different environments, this section demonstrates the procedure to make up a group of clusters after which information of devices is registered in the authentication server, the CA. It will happen just one time before performing the authentication among devices. If another authentication is needed from any different group registration and cluster formation must be done again.

Registration Process: A registration process, in which devices are registered to the server CA, is executed only once. Once the devices are registered to the server, the device and index information are generated and stored on the server. In this way, the authentication and identification of devices are conducted without the need to registrate again. The overhead between the server and the device can be reduced when this method is used. The role of the CA is different from the traditional PKI model. The CA performs a prior arrangement of an adequate key in an off-line node and exerts reliable authority for generated devices. The certification of the server CA verifies the status information of the public key's certification process in real-time using the Online Certificate Status Protocol. D. In the registration process, operator ← means substitution and = means verification.

1) After device A selects the device identity information DI_A, password $Pass_A$ and a random number $r_A^i \in$ [2, n-2] of the owner of device A, device A generates an AID_A value via h($DI_A \oplus r_A^i$) || h($DI_A \oplus Pass_A$) || DI_A. Device A encrypts the ID_A value, an AID_A value and r_A^i value using the public key of the server CA and transmiting it to the server CA.

$$\text{Select } DI_A,\ Pass_A,\ r_A^i \tag{1}$$

$$AID_A \leftarrow \text{h}(\ DI_A \oplus r_A^i\ ) \parallel \text{h}(\ DI_A \oplus Pass_A) \parallel DI_A \tag{2}$$

$$PE_{PU_{CA}}(\text{h}(\ AID_A \oplus ID_A\ )\| r_A^i\ ), ID_A \tag{3}$$

2) The server CA saves the decryption value, which is the decrypted transmitted value. The server CA selects a random number $r_{CA}^i \in$ [2, n-2] and computes a static public key pair (q_{CA}, Q_{CA}). In order to send the server CA information to device A, the server CA generates a $AID_A^{'}$ value that hashes AID_A and r_{CA}^i. The CA chooses secure one-way hash functions $H(\cdot) : \{0,1\} \rightarrow G_P$ and computes $AK_A = q_{CA} \cdot H_1(\ ID_A\ ) \in G_P$, where AK_A is the authentication key for device A and G_P is a cyclic addition group that is generated by P over E(F_q). ID_A is required for a CA to calculate device A's authentication key as shown in Equation (7), i.e., $AK_A = q_{CA} \cdot H_1(\ ID_A\ )$. A one-way hash function $H_1(\cdot) : \{0,1\} \rightarrow G_P$ is applied to ID_A. Then, the CA sends AK_A to device A in a secure channel.

$$PD_{PR_{CA}}(\ PE_{PU_{CA}}(\ h\ (\ AID_A \oplus\ ID_A\ )\| r_{CA}^i\ )) \tag{4}$$

$$Select\ \ r_{CA}^i \tag{5}$$

$$AID_A^{'} \leftarrow h\ (\ AID_A \oplus\ DI_A\ ) \parallel h\ (\ AID_A \oplus\ r_{CA}^i\ ) \parallel ID_A \tag{6}$$

$$AK_A = q_{CA} \cdot H_1(\ ID_A\ ) \in G_P \tag{7}$$

3) The CA transmit the information of public/private pairwise keys to device A through a secure channel. After the computation of public/private pairwise keys of device A, the server CA computes the value of a secret symmetric key of device A. The server CA sends a message encrypted using device A's public key, denoted as $PE_{PU_A}(h\ (\ AID_A^{'} \oplus\ r_{CA}^i\ ),\ AK_A,\ DI_A)$, as well as a message encrypted using device A's random key, denoted as $PE_{r_A^i}(h\ (\ AK_A \oplus\ AID_A\ ),\ AID_A,\ r_{CA}^i\ )$, to device A over a secure channel. That is, in addition to $AID^{'}$, device A receives a random key r_{CA}^i created and encrypted by the CA using device A's public key PU_A and device A's random key r_{CA}^i. Device A can decrypt the received encrypted messages using the random key r_{CA}^i that device A itself owns.

$$PE_{PU_A}(h\ (\ AID_A^{'} \oplus\ r_{CA}^i\ ),\ AK_A,\ DI_A),\ PE_{r_A^i}(h\ (\ AK_A \oplus\ AID_A\ ), AID_A,\ r_{CA}^i\ ) \tag{8}$$

4) Device A receives $h(AID_A^{'} \oplus r_{CA}^{i})$, AK_A, DI_A using a private key PD_{PR_A} and $h(AK_A \| AID_A)$, AID_A, r_{CA}^{i} using a random key r_A^i of device A with the transmitted value from the server CA. Device A checks if $AK_A \cdot P = Q_{CA} \cdot H_1(ID_U)$ holds. If the equation holds, device A keeps AK_A in private and computes or verifies the integrity of the CA keys. Next, device A hashes the current an AID_A value and verifies $AID_A^{'}$.

$$PD_{PR_A}(PE_{PU_A}(h(AID_A^{'} \oplus r_{CA}^{i}), AK_A, DI_A)), PD_{r_A^i}(PE_{r_A^i}(h(AK_A \oplus AID_A), AID_A, r_{CA}^{i})) \quad (9)$$

$$Check \quad AK_A \cdot P \stackrel{?}{=} Q_{CA} \cdot H_1(ID_U) \quad (10)$$

$$Keep \quad AK_A \quad in \quad private \quad (11)$$

$$h(AID_A) = AID_A^{'} \quad (12)$$

5) After device A verifies $AID_A^{'}$, it sends encrypted $r_{CA}^{i}{}^{'}$ values to the server CA using the public key of the server CA. $r_{CA}^{i}{}^{'}$ denotes the secret key of i^{th} generated of the entity CA.

$$PE_{PU_{CA}}(r_{CA}^{i}{}^{'}), ID_A \quad (13)$$

6) The server CA decrypts the transmitted value and verifies the decrypted information values ($r_{CA}^{i}{}^{'}$) with its current information values. After successfully finishing the verification process, the server CA sends a successful registration message to device A.

$$PD_{PR_{CA}}(PE_{PU_{CA}}(r_{CA}^{i}{}^{'})) \quad (14)$$

$$r_{CA}^{i} = r_{CA}^{i}{}^{'} \quad (15)$$

Server Lookup for Cluster Formation: When a new device wants to join a network, it listens **to** a SN beacon. All SNs are responsible for sending to their clusters SN beacons that contain information regarding the member peer and a SN in their cluster. If the node can receive a SN beacon from any cluster, it means that there is an existing cluster and that the node must join the existing cluster. Otherwise, a new cluster must be formed. Fig. 6 shows the server lookup for cluster formation.

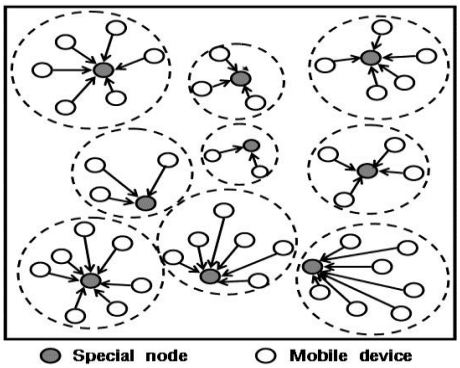


Fig. 6. Server lookup for cluster formation

If a new device x_i does not receive a SN beacon, a new node x_i recognizes that no cluster exists and forms a new cluster. Then the node initiates the task of forming a network. During the process, the node announces itself as a SN by sending SN beacons, provided that the node has a valid certificate sent by a CA whose certificates are valid in that region. Otherwise, the node begins to transmit a member peer beacon to indicate that it wants to be a member node of a network. When the new node enters a network with a valid CA certificate and receives a MN beacon, the node may announce itself as a SN if it desires. Otherwise, it sends member peer beacons and waits for a SN beacon to arrive.

If a new device x_i receives a SN beacon, the new device x_i that receives the SN beacon from a specific region C_i must prove its trustworthiness to the region. If the node has a certificate from the CA of that region, the node submits its public key along with a digitally signed certificate from the CA of that region. As C_i itself is a trusted source certified by the same CA, it has the public key of the CA; hence, it can easily verify the genuineness of x_i. Upon verifying the genuineness of x_i, C_i generates certificates CERT (x_i, C_i), which are signed messages specifying a x_i value and the corresponding public key. Thus, C_i transmits to x_i a message encrypted by the public key of x_i as described below.

3.3.2 Device Authentication Protocol

The proposed protocol is the key allocation among devices to perform the authentication among devices within a same group. Next it is necessary to format SA safely with session key generated and to transmit/receive data. The authentication among devices is performed whenever communication is needed among devices. The SA establishment stops as soon as the communication is terminated. These procedures are helpful in heterogeneous environments.

Key Distribution Process: The key distribution process illustrates the manner in which device A can authenticate device B. It shows that the proposed procedure is more efficient in terms of the processing steps and that it is safer in terms of security than current techniques. Not only can it verify the security of a message by measuring the T value of the message, but it also uses the authentication key from the session key generated from each device. In the key distribution process, substitution is denoted by ← and verification by =.

1) Device A selects a random secret key, d_A and generates a public key, Q_A. It is essential to be able to pick points P uniformly and randomly on an elliptic curve $E(F_q)$ in probabilistic polynomial time. Device A randomly chooses a point $Q_A = (x_A, y_A) \in E(F_q)$, where x_A and y_A are x and y coordinates point Q_A, respectively. If an element $x_A \in F_q$ is the x-coordinate of some point in $E(F_q)$, then we can find y_A such that $(x_A, y_A) \in E(F_q)$ by solving a root finding problem in F_q.

In order to generate Q, device A computes $E_1 = h(x_A \oplus y_A)$. Then, device A computes $t_1 = H_2(T_1)$, $AK_A = q_A \cdot H_1(ID_B)$, $M_A = Q_A + t_1 \cdot AK_A$ and $\overline{Q}_A = x_A \cdot P$, where T_1 is a timestamp that denotes the current time. Finally, device A sends $E_{h(DI_A)}(Q, Q_A, ID_A, M_A, \overline{Q}_A)$, ID_A, T_1 to device B.

$$Q_A = q_A \cdot G = (x_A, y_A) \tag{16}$$

$$Select \ q_A, Q_A \tag{17}$$

$$Compute \ E_1 = h(x_A \oplus y_A) \tag{18}$$

$$Q \leftarrow h(h(DI_A) \oplus E_1) \tag{19}$$

$$t_1 = H_2(T_1) \in Z_P^* \tag{20}$$

$$AK_A = q_A \cdot H_1(ID_B) \tag{21}$$

$$M_A = Q_A + t_1 \cdot AK_A \tag{22}$$

$$\overline{Q}_A = x_A \cdot P \tag{23}$$

$$E_{h(DI_A)}(Q, Q_A, ID_A, M_A, \overline{Q}_A), ID_A, T_1 \tag{24}$$

After receiving $E_{h(DI_A)}(Q, Q_A, ID_A, M_A, \overline{Q}_A)$, ID_A, T_1, device B computes $AK_A' = q_B \cdot H_1(ID_A)$, $t_1 = H_2(T_1) \in Z_P^*$ and $Q_A' = M_A - t_1 \cdot AK_A' = (x_A', y_A')$ to obtain Q_A'. Then, device B checks if $\overline{Q}_A = x_A' \cdot P$ holds. If the equation holds, device B confirms that device A is valid and $x_A' = x_A$. Otherwise, the protocol is terminated.

$$AK_A' = q_B \cdot H_1(ID_A) \tag{25}$$

$$t_1 = H_2(T_1) \in Z_P^* \tag{26}$$

$$Q_A' = M_A - t_1 \cdot AK_A' = (x_A', y_A') \tag{27}$$

$$Check \ \overline{Q}_A \stackrel{?}{=} x_A' \cdot P \tag{28}$$

2) Device B computes $E_1' = h(x_A' \oplus y_A')$ using the transferred Q_A' value and then generates a $h(h(DI_A) \oplus E_1')$ value that compares with Q. If successful, device B is assured of the user DI_A value.

$$E_1^{'} = h\,(x_A^{'} \oplus y_A^{'}) \tag{29}$$

$$Q \leftarrow h(h(DI_A) \oplus E_1^{'}) \tag{30}$$

3) Device B selects a secret key, q_B and generates a public key, Q_B. device B randomly chooses a point $Q_B = (x_B, y_B) \in E(F_q)$, and then it computes $t_2 = H_2(T_2) \in Z_P^*$. Then Device B computes the $M_B = Q_B + t_2 \cdot AK_A^{'}$. Device B computes $E_2 = h(x_B \oplus y_B)$. In order to generate $Q^{'}$ and $T^{'}$, device B computes $h(h(DI_B) \oplus E_2)$ and $h(h(DI_B) \oplus Pass_B \oplus E_2)$. Finally, device B computes $\overline{Q}_B = x_B \cdot P$ and $SK = (H(x_A^{'}, x_B) + x_B) \cdot P$ and sends $E_{h(DI_B)}(Q^{'}, Q_B, M_B, SK, ID_B, \overline{Q}_B), ID_B, T_2$ to device A.

$$Generate\ Q_B = q_B \cdot G = (x_B, y_B) \in E(F_q) \tag{31}$$

$$Select\ \ q_B, Q_B \tag{32}$$

$$t_2 = H_2(T_2) \in Z_P^* \tag{33}$$

$$M_B = Q_B + t_2 \cdot AK_A^{'} \tag{34}$$

$$Compute\ E_2 = h(x_B \oplus y_B) \tag{35}$$

$$Q^{'} \leftarrow h(h(DI_B) \oplus E_2) \tag{36}$$

$$T^{'} \leftarrow h(h(DI_B) \oplus Pass_B \oplus E_2) \tag{37}$$

$$\overline{Q}_B = x_B \cdot P \tag{38}$$

$$SK = (H(x_A^{'}, x_B) + x_B) \cdot P \tag{39}$$

$$E_{h(DI_B)}(Q^{'}, Q_B, M_B, SK, ID_B, \overline{Q}_B), ID_B, T_2 \tag{40}$$

4) After receiving $E_{h(DI_B)}(Q^{'}, Q_B, M_B, SK, ID_B, \overline{Q}_B), ID_B, T_2$, device A computes $t_2 = H(T_2) \in Z_P^*$ and $Q_B^{'} = M_B - t_2 \cdot AK_A = (x_B^{'}, y_B^{'})$ to derive $\overline{Q}_B^{'} = x_B^{'} \cdot P$. Then, device A checks if $\overline{Q}_B^{'} = x_B^{'} \cdot P$. If the equation holds, device S computes $E_2^{'} = h\,(x_B^{'} \oplus y_B^{'})$. Otherwise, the protocol is terminated. After checks $Q_B^{'}$, device A computes session key $SK^{'} = (H(x_A, x_B^{'}) + x_B^{'}) \cdot P$ and substitutes T for $h(h(DI_B \oplus Pass_B) \oplus E_2^{'})$. Then, device A checks if $SK = SK^{'}$ holds and if $T = h(h(DI_B \oplus Pass_B) \oplus E_2)$ holds. If the equation holds, device A confirms that device B is valid. Otherwise, the protocol is terminated.

$$t_2 = H(T_2) \in Z_P^* \tag{41}$$

$$Q_B^{'} = M_B - t_2 \cdot AK_A = (x_B^{'}, y_B^{'}) \tag{42}$$

$$Check\ \ \overline{Q}_B^{'} \stackrel{?}{=} x_B^{'} \cdot P \tag{43}$$

$$Compute\ E_2^{'} = h\,(x_B^{'} \oplus y_B^{'}) \tag{44}$$

$$SK' = (H(x_A, x_B') + x_B') \cdot P \tag{45}$$

$$Substitute \quad T \leftarrow h(h(DI_B \oplus Pass_B) \oplus E_2') \tag{46}$$

$$Check \quad SK = SK' \tag{47}$$

$$Check \quad T \stackrel{?}{=} h(h(DI_B \oplus Pass_B) \oplus E_2) \tag{48}$$

If the process succeeds, device A transmits $h(Q_B' \oplus ID_A)$, $PE_{PU_A}(E_2' \oplus ID_A)$, ID_A to device B.

5) Device B verifies the transmitted $h(Q_B' \oplus ID_A)$, $PE_{PU_A}(E_2' \oplus ID_A)$, ID_A value. Device B compares $Q_B = Q_B'$ and $E_2 = E_2'$ and then checks $T' = h(h(DI_A \oplus Pass_A) \oplus E_A')$. If the equation holds, device B confirms that device A is valid. Otherwise, the protocol is terminated.

$$h(Q_B' \oplus ID_A), PE_{PU_A}(E_2' \oplus ID_A), ID_A \tag{49}$$

$$Compare \quad Q_B = Q_B' \tag{50}$$

$$Compare \quad E_2 = E_2' \tag{51}$$

$$Check \quad T' \stackrel{?}{=} h(h(DI_A \oplus Pass_A) \oplus E_A') \tag{52}$$

SA Establishment Process: The security association (SA) is a shared state involving a cryptographic key, the identity of the other side, a sequence number, and the cryptographic algorithms to be used. It is used for carrying on a cryptographical protected conversation. Associated with each device of the SA is a cryptographic key and other information such as the identity of the other device, the sequence number currently being used, and the cryptographic services being used (e.g., integrity only, or encryption+integrity, as well as which cryptographic algorithms should be used). In the SA establishment procedure, the SA is considered unidirectional; thus, a conversation between device A and device B will consist of two SAs, one in each direction. The PDs (Personal Databases) of each device have no communication record regarding the corresponding device. Therefore, when a SA is established between arbitrary devices, a communication record is generated in the secret table of each device. The secret table in each PD is the core of the security protocol. If it is compromised, all security related to the device is compromised. In detail, establishing the SA operates as follows: Device A sends a communication start request message to device B. Device B has communicated with the server CA in the new node-addition procedure, hence the communication record between the server CA and device B has been stored in each PD. The server CA sends a response message to device B over the established SA. This message contains arbitrary communication between record numbers and related data for mutual authentication between devices A and B.

As the authentication information between server CA and device B has already been stored in device B's PD, it is not necessary to send this information to device B. Server CA authenticates device B using the proposed mutual authentication method using authentication records. Device B generates an authentication key from the secret key using the two types of authentication information and a random number. It then sends an authentication request

message to device A. Server CA authenticates device A via the proposed mutual authentication method using authentication records. The following procedure at device A is identical to the procedure of device B: PD retrieval, mutual authentication with the server CA, acquisition of the authentication data between the server CA and device B, and generation of the secret/authentication key. Mutual authentication is executed by each node; if successful, a SA is established between device A and device B.

4. Analysis and Evaluation

In this section, we describe the simulations of mobile P2P topologies with a DSS clip trace [26] and mobile P2P topologies that are dynamic node changes including joining and leaving procedures by assigning a lifetime in seconds to every node. Each device can make a random movement so that the overlay topologies change accordingly.

4.1 Environment

In analyzing the performance of the proposed protocol, the parameters and storage space requested for a mobile device are defined as follows: In the mobile device memory, a total of 96 bytes are required to save the fixed public/private pairwise keys of the device node and implicated certificate. Implicated certificate means that only device A and device B, and the participants who both of the devices trust know the key.

Reduced storage space in the ROM is as following: 200 bytes for the generation and verification of an ECC implicated certificate, 630 bytes for a modular p integer library, 790 bytes for a general library, 510 bytes for SHA-1, 1 Kbytes for the AES symmetric key algorithm, 1,400 bytes for previously computed data, 20 bytes for the base p phase, and 20 bytes for elliptic order n.

4.2 Basic Operation Costs

To illustrate the operation cost for each signature/verification scheme, we use the same method used in [28]. Table 2 shows the measurements for the generation of the signature/verification data using plain RSA and DSA, respectively.

Table 2. Signature/verification Costs of Basic Schemes in msecs (P4 1.8GHz)

Key	Milliseconds/ Operation	Megacycles/Operation
RSA 1024 Signature	1.42	2.60
RSA 1024 Verification	0.07	0.13
RSA 2048 Signature	5.95	10.89
RSA 2048 Verification	0.15	0.28
DSA 1024 Signature	0.47	0.85
DSA 1024 Signature with precomputation	0.41	0.76
DSA 1024 Verification	0.52	0.95
DSA 1024 Verification with precomputation	0.66	1.21

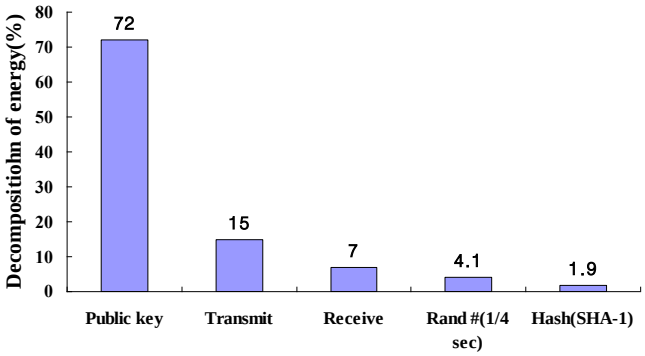


Fig. 7. Decomposition of energy for mutual authentication and key exchange on the mobile device side

Fig. 7 displays the energy consumption ratio, in which the energy consumed by devices for mutual authentication and key exchanges are divided into those of public key, transmission, receiving, random, and hash values. The public key used for the ECC amounts to 72% of the total energy consumption, and communication costs consume 22% of the total energy. Random number generation and hash costs represent 4.1% and 1.9%, respectively, which are values that can be ignored in terms of the total energy consumption.

4.3 Traffic Costs

The experimental results of the traffic overhead and response time of DAMMA are shown in Fig. 8. The traffic cost is one of the most important parameters that network administrators consider. The traffic cost exerted by DAMMA is mainly due to share flooding. Indeed, the more the shares are split, the higher the traffic cost.

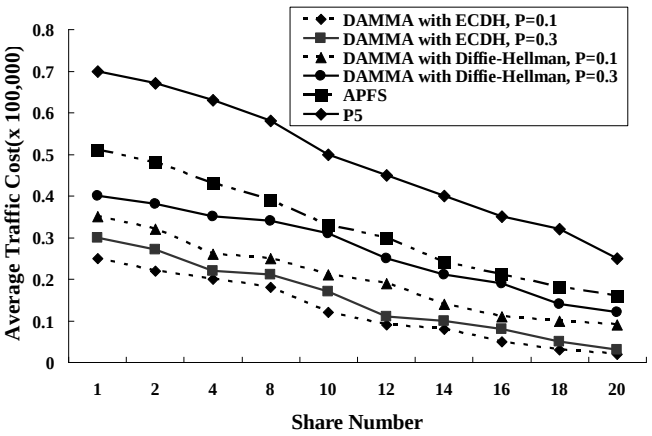


Fig. 8. Traffic Cost of Mobile Device

When the number of split shares is 10 and the probability is 0.3, the average traffic cost is 23,103, which is close to 24,368, the average normal flooding traffic cost of the system. Although increasing the average probability leads to a high query recovery rate from shares, the traffic cost grows as well.

4.4 Communications Costs

A mobile device of DAMMA requires computation costs such as one random point EC scalar product, one fixed point EC scalar product, two symmetric encryption and decryption functions, four key hash functions, one hash function, and one random number generation. SHA-1 requires only 2 ms to decrease to 128 bits in the binary string on M16C microprocessor alone. For symmetric encryption and decryption, the AES block cipher is hypothesized as 256 bits of a text block and is used to create keyed-hash functions. The response time between mobile devices until the source node receives the first response of the query after sending a query to the destination node is estimated. Fig. 9 indicates that the additional time resulting from the proposed scheme is approximately 15%-18% more than the normal query response time.

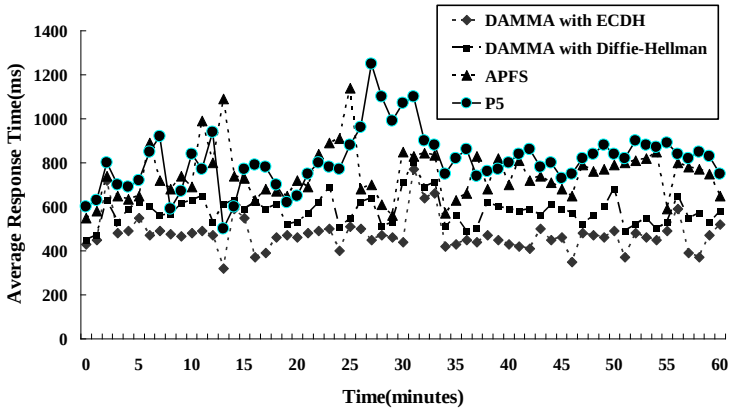


Fig. 9. Query Response Time of Mobile Device

Agent peers that recover original queries may incur a large number of replicate messages into the system by flooding the query. The number of agent peers mainly depends on the average flooding probability and the threshold of the proposed scheme. To guarantee that a sufficient number of (but not too many) agent peers are involved in the query flooding process, the initiator peer should carefully choose the threshold of the proposed scheme according to the flooding probability. Fig. 10 indicates that in most cases, the flooding probability p is less than 0.5; thus, individual peers can choose the proper threshold accordingly.

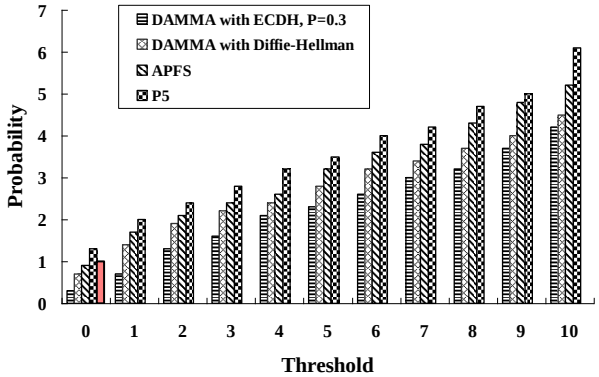


Fig. 10. Threshold and number of agent peers

4.5 Storage Costs

The performance evaluation of the storage cost is based on the degree of cryptographic operation complexity, the time for performance, the amount of data, the total amount of data transmitted for the operation of each protocol, and the memory requirements. In this paper, according to the rekeying the costs in the communication operation, the storage costs for the CA and for mobile devices are computed below, with n representing the number of mobile devices and r the range of a mobile device. The storage of the mobile device key is the number of keys per user: $\log_2 \frac{N}{M} + \log_2 M = \log_2 N$. The storage of the CA key is the number of keys: $(2 \times \frac{N}{M}) + \frac{N}{M} \times (2 \times \log_2 M) = 2 \times \frac{N}{M} (\log_2 M + 1) - 1$. The communication cost for rekeying is $2 \times \log_2 \frac{N}{M} + \log_2 M = 2 \times \log_2 N - \log_2 M$. The highest value of M is the lowest value of the rekeying cost and the CA key storage. The non-volatile (FLASH) memory required for the mobile device requires 96 bytes, including the public/private pairwise keys (q_S, Q_S) and the implicated certificate. In addition, program ROM requires 4.8 Kbytes for code and data, including the previously computed table above, along with the order and prime number of the elliptic curve, the AES symmetric key algorithm, SHA-256, the integer library, the modular operation, and the generation of implicated ECC certificates.

4.6 Scalability

To determine the expandability in the P2P environment, the change in response time and traffic overhead can be checked in general as the P2P overlay increases [32]. In Fig 11, the expandability of the DAMMA protocol is compared with that of the conventional protocol through the change of response time according to the increase of the overlay size. The size of the P2P overlay in the experiment for the comparison was set as 1,000-10,000, and the average response time was measured for 1,000 queries. The DAMMA protocol showed a 3% lower collision than the conventional protocol in the P2P overlay increase. This is because the DAMMA protocol has lower costs than other protocol in traffic, communication and storage. Furthermore, with the relatively higher anonymity that becomes the basis of the DAMMA protocol, it has a better expandability.

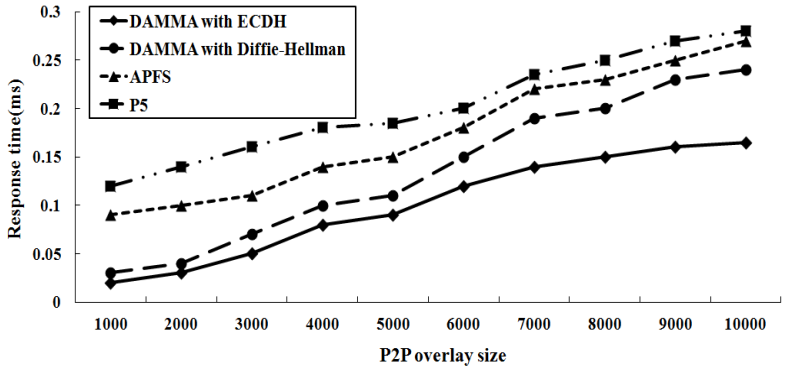


Fig. 11. Scalability of proposed protocol

5. Security Efficiency

5.1 Replay Attacks

In the proposed scheme, a replay attack fails because the authenticity of data transmitted in mutual authentication of devices A and B can be verified using time stamps T_1 and T_2, random points $Q_A^{'}$ and $Q_B^{'}$, and the shared session key SK. Only devices A and B that have access to the session key SK and authentication keys AK_A can include values x_A, x_B, $x_A^{'}$, and $x_B^{'}$ in hashed messages $E_1, E_1^{'}, E_2, E_2^{'}$, so the proposed scheme is resistant to replay attacks.

5.2 Impersonation Attacks

Suppose that an attacker impersonating device A (a legitimate device) takes $Q_A^* = q_A \cdot G = (x_A^*, y_A^*)$ and AK_A^* in order to compute $M_A^* = Q_A^* + t_1 \cdot AK_A^*$. For authentication, the attacker sends $E_{h(DI_A)}(Q^*, Q_A^*, ID_A, M_A^*, \overline{Q}_A^*)$, ID_A and T_1 to device B. In device B, M_A^* is computed using AK_A^* instead of AK_A, so $AK_A^* = (x_A^*, y_A^*)$ is not derived from $Q_A^{'} = M_A^* - t_1 \cdot AK_A^*$. Because $\overline{Q}_A$ is not equal to $x_A^{'} \cdot P$ ($\overline{Q}_A^{'} \neq x_A^{'} \cdot P$), device B will notice that an impersonation attack has taken place. Similarly, the impostor does not know device B's secret key q_B, so he/she cannot impersonate device B either.

5.3 Stolen-verifier Attacks

The proposed scheme is secure against stolen verifier attacks because device B does not store the user's verifier (e.g., hashed passwords). In the proposed scheme, the authentication key AK_A that authenticates device A is computed using device A's identifier ID_A and CA's secret key q_{CA}, i.e., $AK_A = q_{CA} \cdot h(ID_A)$. The CA maintains its secret key q_{CA} regardless of device A's value. When a new device is added, the CA is not required to store the added device's verifier (password or public key) into its database. Hence, the proposed scheme is protected against stolen verifier attacks and provides high scalability. Enhanced scalability regarding adding network nodes makes the proposed scheme suitable for applications with a large number of devices.

5.4 Session Key Security

Session key security indicates that the communicating devices (the sender and recipient) are in possession of a shared session key that only they are familiar with. A shared session key SK is created using parameters x_A and x_B, and these parameters are different in each session. Only devices A and B (i.e., communicating devices) gain access to the parameters to ensure the security of a communication session. A session key is a single-use symmetric key used for encrypting and decrypting all messages in one communication session, so the session key becomes obsolete when the communication session is finished. When device A and B enter a new session, a new session key is created and all messages transmitted are encrypted with the new session key. Even if an attacker gets a copy of the previous session keys, the obtained session keys cannot decrypt the encrypted messages of the current session, including the current session key SK encrypted using the recipient's public key. The earlier session keys

$\overline{Q}_A$ and $\overline{Q}_B$, even if they have been exposed, are still randomly generated keys, so they cannot compute the next session key. In addition, using a long random number as the session key reduces the risk that an attacker could simply guess a valid session key through trial and error or brute force attacks. Therefore, the proposed scheme provides session key security.

5.5 Perfect Forward Secrecy

In authenticated key-agreement protocols that use public key cryptography, perfect forward secrecy is required of the property that a session key derived from a set of long-term public and private keys will not be compromised; even if one of the long-term secrets (i.e., private keys) is compromised in the future. In the proposed scheme, an attacker cannot compute an earlier session key SK even if device A's authentication key AK_A derived from the same long-term keying material as $\overline{Q}_A$ in a subsequent run is compromised. Hence, the proposed scheme provides perfect forward secrecy, and the security of the session keys legitimately created in the previous sessions is guaranteed.

5.6 Anonymity

For two reasons, the DAMMA protocol uses a cryptological hash function such as SHA-1 to change a real identity into an anonymous identity AID. First, the hash function is efficient and easily usable. Second, the hash function can be used openly by all the devices in all networks without exchange of secret information that is shared between devices. These properties are extremely appropriate for the open P2P environment, with which the DAMMA protocol can secure anonymity.

The anonymity of device identity comes directly from the one-way property of the cryptological hash function. Suppose that the hash function $h(\cdot)$ used in the DAMMA protocol is a well designed function with an m-bit-long hash value, and there is no defect in cryptanalysis. The $h(\cdot)$ has a strong resistance against the pre-image-resistance property, which means it is infeasible to find x value for a given y value in $h(x) = y$. Here, 'infeasible' means that the evaluation calculation of the hash function should be carried out at least 2^{m-1} times in general to determine x [30]. In order to make the two devices have the same AID, some malicious device can launch an improved attack to discover two different identities even though they are not the real identities. By using one of the two device identities, the malicious device will try to disguise itself as a normal device, but such an attack can be withstood by the collision-resistance of the hash function, for it is infeasible to find (x, y) pairs in calculation as $h(x) = y$. By the hash function with an m-bit hash value, the infeasible $2^{m/2}$ calculation for m≥128 is the same as to require finding the collision with 1/2 probability.

The reason why the one-way hash function is used in the DAMMA protoci is for an attacker not to use the disclosed information of the pre-image. The attacker tries to restore AID after recording the proper service transacton or trace the device after recording the random number, r_{CA}^{i}, in order to generate AID. But the DAMMA protocol periodically changes the random number, r_{CA}^{i}, for AID not to be calculated. Since it is difficult for the hash function to be transformed inversely, if the attacker captures the output of device identification, the device identification could be secure. The security is also ensured while the communication between the CA and the device is eavesdropped, because the identification of the new CA is encoded with the previous CA when the device updates the identification information of a new CA on its memory.

To determine the pre-image of the hash value for M=64, it requires 600,000 mips-years while 1 mips-h is necessary to calculate the 2^{32} collisions of the hash value. Here, the hash value length should be longer than 64 bits. In the DAMMA protocol, 'm' must be greater than 90 for enhanced safety and efficiency. Therefore, the hash function using SHA-1(m=160) is appropriate for the DAMMA protocol. If SHA-1 is selected in the DAMMA protocol, the malicious device will have to carry out 2^{159} calculations to determine the identity from the AID and 2^{80} calculations to determine a pair with the same output value as the input value. Thus, the DAMMA protocol perfectly supports the anonymity for the device in a P2P network. No malicious device can infer the real identity from the AID of the DAMMA protocol. Simply, the DAMMA protocol provides the function to convert a real identity to an anonymous AID safely.

5.7 Tracebility(Conditional Anonymity)

If an arbitrary device abuses AIDs in the DAMMA protocol, the authority of the device revokes the identity of the arbitrary device abusing the AID through the cooperation of other devices and the SN. In the DAMMA protocol, the use of session key SK between devices provides the role to restrict the abuse of anonymity. If the real identity of a device is exposed to other devices in the registration process before the generation of session key SK, the device that does not use the DAMMA protocol cannot extract the exposed information without the session key SK. Even though the malicious device acts anonymously, it causes no problem because the information is kept diversely between the SN and the normal device. If it is necessary to limit the authority of a malicious device, the DAMMA protocol can trace the identity of abnormal device as shown in [31].

6. Conclusions

Providing a reliable and efficient authentication protection among peers is highly desirable in order to build a scalable and secure P2P system. This paper presents the DAMMA protocol, which is suitable for mutual authentication in mobile P2P networks. After several hop-to-hop requests, the proposed protocol broadcasts a request that is normally a small message. It then sends back the requested file to the mobile device not via broadcasting, but through a dynamically created covert path to achieve both communication authentication and efficiency. The proposed DAMMA protocol is based on the standard ECDH key establishment protocol. In order to minimize the costs involved with scalar multiplications, the protocol uses a temporarily encrypted channel to protect the randomization of the established pairwise keys. The DAMMA protocol is more secure against known-key security attacks than symmetric-key-based protocols, because it does not assume protection of the nodes during key bootstrapping periods. Moreover, corrupted or captured nodes cannot perform impersonation, sybil or fake generation attacks on any node other than the corrupted one. We are exploring the approaches combining different methods to further synergistically achieve the goal of both strong authentication and high communication efficiency, as well as to adapt to application needs and network conditions.

References

[1] K. Bennett and C. Grothoff, "GAP - Practical anonymous networking," in *Proc. of Privacy Enhancing Technologies workshop*, 2003. http://dx.doi.org/10.1007/978-3-540-40956-4_10

[2] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong, "Freenet: A Distributed Anonymous Information Storage and Retrieval System," in *Proc. of Workshop on Design Issues in Anonymity and Unobservability*, Berkeley, CA, USA, 2000. Article (CrossRef Link)

[3] R. Dingledine, M. J. Freedman, and D. Molnar, "The Free Haven Project: Distributed Anonymous Storage Service," in *Proc. of Workshop on Design Issues in Anonymity and Unobservability*, 2000. PMid:10736377. Article (CrossRef Link)

[4] D. Kugler, "An Analysis of GNUnet and the Implications for Anonymous, Censorship-Resistant Networks," in *Proc. of Privacy Enhancing Technologies workshop*, 2003. http://dx.doi.org/10.1007/978-3-540-40956-4_11

[5] A. Serjantov, "Anonymizing Censorship Resistant Systems," in *Proc. of the First International Workshop on Peer-to-peer Systems*, 2002. http://dx.doi.org/10.1007/3-540-45748-8_11

[6] M. Waldman, A. D. Rubin, and L. F. Cranor, "Publius: A Robust, Tamper-evident, Censorship-resistant Web Publishing System," in *Proc. of the 9th USENIX Security Symposium*, 2000. PMCid:2560618. Article (CrossRef Link)

[7] S. Sherwood, B. Bhattacharjee, and A. SrinivasanM, " P^5 : A Protocol for Scalable Anonymous Communication," in *Proc. IEEE Symp. Security and Privacy*, May 2002. http://dx.doi.org/10.1109/SECPRI.2002.1004362

[8] J. Han, Y. Liu, L. Xiao, R. Xiao and L. M. Ni "A Mutual Anonymous Peer-to-peer Protocol Design," in *Proc. of the 19th IEEE International Parallel and Distributed Processing Symposium*, 2005. http://dx.doi.org/10.1109/IPDPS.2005.49

[9] L. Xiao, Z. Xu, and X. Shang, "Low-cost and Reliable Mutual Anonymity Protocols in Peer-to-Peer Networks," *IEEE Transactions on Parallel and Distributed Systems*, 2003. http://dx.doi.org/10.1109/TPDS.2003.1233706

[10] V. Scarlata, B. N. Levine, and C. Shields, " Responder Anonymity and Anonymous Peer-to-Peer File Sharing," in *Proc. of the 9th International Conference of Network Protocol(ICNP)*, 2001. PMCid:1301537. http://dx.doi.org/10.1109/ICNP.2001.992907

[11] X. Liu, L. Xiao, A. Kreling, and Y. Liu, "Optimizing Overlay Topology by Reducing Cut Vertices," in *Proc. of ACM NOSSDAV*, 2006. http://dx.doi.org/10.1145/1378191.1378213

[12] X. Wang, S. Chellappan, P. Boyer, and D. Xuan, "On the Effectiveness of Secure Overlay Forwarding Systems under Intelligent Distributed DoS Attacks," *IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS*, 2005. http://dx.doi.org/10.1109/TPDS.2006.93

[13] Y. Liu, A. -H. Esfahanian, L. Xiao, and L. M. Ni, "Approaching Optimal Peer-to-Peer Overlays," in *Proc. of the 13th Annual Meeting of the IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems(MASCOTS)*, 2005. http://dx.doi.org/10.1109/MASCOTS.2005.15

[14] X. Y. Zhang, Q. Zhang, Z. Zhang, G. Song, and W. Zhu, "A Construction of Locality-Aware Overlay Network:mOverlay and Its Performance," *IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS*, 2004. http://dx.doi.org/10.1109/JSAC.2003.818780

[15] D. Qiu and R. Srikant, "Modeling and Performance Analysis of BitTorrent-Like Peer-to-Peer Networks," in *Proc. of ACM SIGCOMM*, 2004. Article (CrossRef Link)

[16] Y. Liu, Z. Zhuang, L. Xiao, and L. M. Ni, "A Distributed Approach to Solving Overlay Mismatch Problem," in *Proc. of the 24th International Conference on Distributred Computing Systems(ICDCS)*, 2004. http://dx.doi.org/10.1109/ICDCS.2004.1281576

[17] X. Liu, Y. Liu, and L. Xiao, "Reliable Response Delivery in Peer-to-Peer System," in *Proc. of the 12th Annual Meeting of the IEEE International Symposium on Modeling Analysis, and Simulation of Computer and Telecommunication Systems(MASCOTS)*, 2004. http://dx.doi.org/10.1109/MASCOT.2004.1348298

[18] K. V. Nguyen, "Simplifying Peer-to-Peer Device Authentication Using Identity-Based Cryptography," in *Proc. of International conference on Networking and Services(ICNS06)*, 2006. http://dx.doi.org/10.1109/ICNS.2006.101

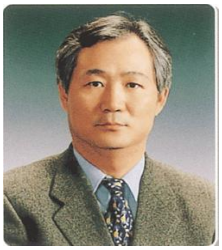
[19] SafeNet, "SafeNet Authentication Manager," SafeNet, Inc., 2013. Article (CrossRef Link)

[20] L.E. Holmquist, F. Mattern, B. Schiele, P. Alahuhta , M. Beigl and H. W. Gellersen, "Smart-Its Friends: A Technique for Users to Easily Establish Connections between Smart Artefacts," in *Proc. Of UBICOMP 2001*, Atlanta, GA, USA, Sept. 2001. Article (CrossRef Link)
[21] T. Kindberg and K. Zhang, "Context authentication using constrained channels," *HP Labs Tech. report HPL-2001-84*, 2001. http://dx.doi.org/10.1109/MCSA.2002.1017481
[22] Y. Liu, X. Liu, L. Xiao, L. M. Ni, and X. Zhang, "Location-aware topology matching in P2P Systems," in *Proc. of IEEE INFOCOM*, 2004. http://dx.doi.org/10.1109/INFCOM.2004.1354645
[23] W. Jia, D. Xuan, W. Tu, L. Lin, and W. Zhao, "Distributed admission control for anycast flows," *IEEE Transactions on Parallel and Distributed Systems(TPDS)*, 2004. http://dx.doi.org/10.1109/TPDS.2004.34
[24] K. P. Gummadi, R. J. Dunn, S. Saroiu, S. D. Gribble, H. M. Levy, and J. Zahorjan, "Measurement, Modeling, and Analysis of a Peer-to-Peer File-Sharing Workload," in *Proc. of the 19th ACM Symposium on Operating Systems Principles(SOSP)*, 2003. Article (CrossRef Link)
[25] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker, "Making Gnutella-like P2P Systems Scalable," in *Proc. of ACM SIGCOMM*, 2003. Article (CrossRef Link)
[26] The Gnutella Protocol Specification v0.4, "http://www.clip2.com/GnutellaProtocol04.pdf". Article (CrossRef Link)
[27] S. Zhu, S. Setia, S. Jajodia, "LEAP: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks," in *Proc. of the Seventh IEEE International Symposium on Multimedia*, 0-7695-2489-3/05 (2005). http://dx.doi.org/10.1145/948109.948120
[28] N. Saxena, G. Tsudik, J. H. Yi, "Admission Control in Peer-to-Peer: Design and Performance Evaluation," in *Proc. of the first ACM Workshop Security of Ad Hoc and Sensor Networks Fairfax*, 2003. http://dx.doi.org/10.1145/986858.986873
[29] J. Han, Y. Liu, L. Xiao, R. Xiao, and L. M. Ni, "A mutual anonymous peer-to-peer protocol design," in *Proc. of 19th IEEE International Parallel and Distributed Processing Symposium*, Denver, CO, United states, 2005. http://dx.doi.org/10.1109/IPDPS.2005.49
[30] B. Schneier, "Applied Cryptography-Protocols, Algorithms, and Source Code in C," *second ed. John Wiley & Sons, Inc.*, 1996. Article (CrossRef Link)
[31] J. Liao, J. Xiao, Y. Qi, P. Huang, and M. Rong, "ID-based signature scheme without trusted PKG," in *Proc. of 1st Conference on Information Security and Cryptology*, Beijing, China, 2005. http://dx.doi.org/10.1007/11599548_5
[32] Li Lu, Jinsong Han, Yunhao Liu, Lei Hu, Jinpeng Huai, Lionel M. Ni, Jian Ma, "Pseudo Trust: Zero-knowledge Authentication in Anonymous P2Ps", *IEEE Transactions on Parallel and Distributed Systems*, Vol. 19, No. 10, October, 2008. http://dx.doi.org/10.1109/TPDS.2008.15
[33] J. S. Han and Y. H. Liu, "Mutual Anonymity for Mobile P2P Systems," *IEEE Transactions on Parallel and Distributed System*, Vol. 19, No. 8, pp. 1009-1019, August, 2008. http://dx.doi.org/10.1109/TPDS.2007.70805
[34] Z. Y. Li, L. M. Wang and S. G. Chen, "Network Coding-Based Mutual Anonymity Communication Protocol for Mobile P2P Networks," in *Proc. of 2012 IEEE 11th International Conference on Communications(TrustCom)*, pp. 986-991, June, 2012. http://dx.doi.org/10.1109/TrustCom.2012.211
[35] Y. H. Liu, J. S. Han and J. L. Wang, "Rumor Riding: Anonymizing Unstructured Peer-to-Peer Systems," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 22, Issue 3, pp. 464-475, Feb. 2011. http://dx.doi.org/10.1109/TPDS.2010.98

Yoon-Su Jeong was born in Cheong-Ju, Korea in 1975. He received the B.S. degree in the department of computer science, Cheongju University in February 1998. He received the M.S. degree and Ph.D in the department of computer science, Chungbuk National University in February 2000 and 2008. He is currently working professor in the department of Information Communication Engineering, Mokwon University. His research interests also include cryptography, network security, information security, AAA, wire/wireless communication security.

Yong-Tae Kim was born in Daejeon, Korea in 1961. He received the B.S. degree in the department of computer science, Hannam University in February 1984. He received the M.S. degree in the department of computer science, Soongsil University in February 1988. He received the Ph.D. degree in the department of computer science, Chungbuk National University in February 2008. He is currently working professor in the department of Multimedia Engineering, Hannam University. His research interests also include mobile web service, network security, information security, AAA and mobile communication security.

Seung-**Soo Shin** was born in Cheong-Ju, Korea in 1966. He received the B.S. degree in the department of mathematic, Chungbuk National University in February 1988. He received the M.S. degree and Ph.D in the department of mathematic, Chungbuk National University in February 1993 and 2001. And He received the Ph.D in the department of computer engineering, Chungbuk National University in February 2004. He is currently working professor in the department of Information Security, Tongmyong University. His research interests also include cryptography, network security, information security.



Sang-Ho Lee was born in PuSan, Korea in 1953. He received the B.S. degree in the department of computer science, Soongsil University in February 1976. He received the M.S. degree and Ph.D in the department of computer science, Soongsil University in February 1981 and 1989 respectively. He is currently working professor in the department of software engineering, Chungbuk National University. His research interests also include Protocol Engineering, Network Security, Network Management and Network Architecture.