Inf Syst Front
DOI 10.1007/s10796-010-9254-0

MILC: A secure and privacy-preserving mobile instant locator with chatting

Athanasios Loukas · Dimitrios Damopoulos · Sofia A. Menesidou · Maria E. Skarkala · Georgios Kambourakis · Stefanos Gritzalis

© Springer Science+Business Media, LLC 2010

Abstract The key issue for any mobile application or service is the way it is delivered and experienced by users, who eventually may decide to keep it on their software portfolio or not. Without doubt, security and privacy have both a crucial role to play towards this goal. Very recently, Gartner has identified the top ten of consumer mobile applications that are expected to dominate the market in the near future. Among them one can earmark location-based services in number 2 and mobile instant messaging in number 9. This paper presents a novel application namely MILC that blends both features. That is, MILC offers users the ability to chat, interchange geographic co-ordinates and make Splashes in real-time. At present, several implementations provide these services separately or jointly, but none of them offers real security and preserves the privacy of the end-users at the same time. On the contrary, MILC provides an acceptable level of security by utilizing both asymmetric and symmetric cryptography, and most importantly, put the user in control of her own personal information and her private sphere. The analysis and our contribution are threefold starting from the theoretical background, continuing to the technical part, and providing an evaluation of the MILC system. We present and discuss several issues, including the different services that MILC supports, system architecture, protocols, security, privacy etc. Using a prototype implemented in Google's Android OS, we demonstrate that the proposed system is fast performing, secure, privacy-preserving and potentially extensible.

A. Loukas · D. Damopoulos · S. A. Menesidou · M. E. Skarkala · G. Kambourakis (✉) · S. Gritzalis
Info-Sec-Lab Laboratory of Information and Communications Systems Security, University of the Aegean,
Samos 83200, Greece
e-mail: gkamb@aegean.gr

Keywords Mobile social networking · Virtual communities · Security · Privacy · Chat · Location-based services · Spatial messaging

Draft

1 Introduction

Today, the advances in wireless communication technologies and the proliferation of mobile devices have enabled the realization of pervasive and intelligent environments for users to communicate with each other, interact with information processing devices, and acquire ubiquitously a plethora of mobile wireless services through various types of access networks. In fact, nowadays the employment of mobile devices such as smart-phones for quick communication and collaboration is almost synonymous to their name. Driven by these factors several categories of consumer mobile applications have emerged. One of them is conceptualized under the general term *mobile social networking*. This can be defined as a special kind of social networking where one or usually a group of individuals sharing similar interests and/or common pursuits, are communicating, conversing and connecting with one another using mobile devices.

Similar to Web based social networking, mobile social networking mandates the existence of a virtual community. According to (Zhong et al. 2008) these virtual communities are being created either by the evolution of existing mobile portals into mobile communities, or by the continuously growing trend for popular Internet social networking tools such as Facebook and MySpace to transmute into mobile, or by services that were designed from the ground up having mobility in mind, like the well known Tweeter and mobile wikis, blikis etc. Normally, such applications may be stand-alone and thus employ a subset of features (e.g., SMS, MMS, instant messaging, buddy-finder, data sharing etc) or interface with an existing web based social networking tool to provide a collection of services. If used properly, mobile networking applications can be proved very useful in several work, educational and/or entertainment spaces promoting interaction, learning, productivity etc, between the members of the community they support. A recent work by (Quercia et al. 2010) is one of many examples of the applications a mobile social network is able to support. The authors showed that by monitoring user's mobile phone activity, i.e., by using Bluetooth or monitoring text messaging and phone calls, the application is capable of a) recommending her possible new friends, b) watching her friendship status with someone (this is done by monitoring how often they communicate or meet), and c) figure out her mood based on her status updates.

In terms of building and promoting virtual communities, the main target of a mobile social networking application should be the facilitation of the user for exchanging information with all the other members of the same community at any time and place. Thus, whatever its usage and the features it offers to the end-user, the need for implementing secure and privacy-preserving mobile social networking applications is considered nowadays more than ever necessary. For instance, in the current version of (Quercia et al. 2010) application described above, all necessary calculations are being carried out on the user's mobile device without having to compromise any user private data. Nevertheless, a future version will carry out all calculations on the server's side where much attention must be given in order to protect user's private sphere. Furthermore, several modern commercial applications support chatting and information exchange in general between their members but they finally turn out to be insecure. Message exchange between members may be in cleartext, the real identity of a participating user can be easily leak out and, in some cases, a user can be tracked and profiled based on her actions and the services she acquires. Even worse, the providers of such services are able to collect and keep for long time detailed log files on users' actions and sometimes sell them to advertising firms for profit. In any case however, a user who participates in a virtual community needs to rest assure that any information she sends and receives remains confidential and that her private sphere is not violated without her consent. A study that was carried out by (Chen and Rahman 2008) highlighted all the above mentioned weakening points in 31 mobile social networking applications designed for Apple's iPhone and underlined the necessity for a design level privacy concern.

In general, privacy is a complex concept that affects aspects such as location, identification and authentication (Askwith et al. 1997). While location privacy requires that the location of a mobile user is untraceable to unauthorized parties (including the network), identification privacy mandates user's anonymity except for authorized parties. As we can perceive, these types of privacy are interrelated. If user's identity remains confidential, then location data is worthless. At the same time, both types of privacy strongly depend on the authentication process where user's permanent identity must be exchanged. If the authentication mechanism does not afford an adequate level of privacy to protect identification related data, the location can be revealed to unauthorized third parties. Under these circumstances, the demand for truly privacy-preserving operation becomes even greater when location-based services come into focus. This is because location information is a set of sensitive data describing an individual's location in real time. If the geographic location of an individual falls into the wrong hands, an adversary can physically locate and possibly track down a person. Therefore, the underlying mechanisms should have the ability to prevent other parties from arbitrarily learning one's current position. Location privacy is about controlling access to this information, which is granted by the user who must be the only one responsible to decide if someone is going to have access to her location data or not.

Contributing to the abovementioned issues we mobilized into making the MILC system (http://milc.samos.aegean.gr). MILC originally started as a mobile social networking application that would give to the members of the University of the Aegean Community (i.e., students, faculty) the opportunity to collaborate with each other. For this reason, MILC is not exactly a typical mobile social network application, but integrates in private or closed communities of scope individuals that participate in the community, mainly for educational reasons (students' communities, research groups etc.). Also, MILC has been designed from the onset to be attractive to young people, lightweight, secure and privacy preserving. These characteristics allow MILC to be straightforwardly useful in a variety of environments with only minor modifications. Currently, the MILC system combines four services i.e., chatting, Buddy-Finder (BF), Points of Interest (PoI) locator, and Spatial Messaging (SM) into one. SM is an advanced wireless networking service that allows users to post a message on a virtual notice board somewhere on the map for someone else to collect. This service is also known with the terms "splash messaging" or "air-graffiti". Of course, the location-based services offered by MILC (i.e., BF, PoI, SM) require the existence of a GPS receiver either built into the device or external. Nevertheless, this is not an issue today because the majority of mobile devices are shipped with a built-in GPS receiver. As already pointed out, MILC contribution is twofold (a) it utilises both asymmetric and symmetric cryptography to provide a high level of security to its users, and (b) it respects end-user privacy by putting the user in control of what private information is revealed to other parties and under what circumstances. Additional features towards strengthening end-users privacy are pseudonymity and the absence of all kind of log files about their actions. MILC prototype has

Draft

been developed using the Google's Android platform and follows the well-known client-server model. In the following, our intension is not only to describe MILC's services and general architecture, but also to analyze and evaluate its major technical parts. In our opinion this would be a major importance for anyone who is interested in building an analogous system and/or extending its functionality.

The rest of the paper is structured as follows. Next section addresses previous work on the topic. Section 3 describes the system's overall architecture and presents and technically analyses major MILC components and underlying protocols. Section 4 evaluates MILC in terms of performance, security and privacy and provides a theoretical comparison with other similar commercial applications. The last session concludes the paper and gives pointers to future work.

2 Related work

In this section we discuss related work. Our analysis considers not only published work in the same topic but also some commercial applications similar to MILC.

Until now, several solutions have been proposed to address the issue of location privacy in proximity detection. The authors in (Ruppel et al. 2006) introduce a solution that applies a distance preserving coordinate transformation to hide the true geographical position of the user in conjunction with pseudonyms to protect user's privacy. However, as discussed in (Liu et al. 2006), the combination of anonymisation techniques with advanced functions like proximity are not appropriate means for protecting end-users privacy as they can be easily attacked. For instance, an attacker is able to recover the original data if he knows the transformed data (in this case the transformed coordinates) and some prior information. That is, a collection of independent samples which may or may not overlap with the original data or a small set of private data and their "perturbed" counterparts. The first method is based on basic properties of linear algebra and the latter on principal component analysis. MILC addresses this issue by transferring the coordinates of any individual in encrypted form between the participating parties (i.e., the client and server).

The work in (Palazzi 2004) proposed a location based application for GSM. The application offers to subscribers an appealing means for entertainment through the provision of multimedia content. It also enables users to be informed about the presence of other subscribers who are registered in the electronic index book of the user's mobile device. This means that a user can find if one of her friends is near. However, this work does not address confidentiality and integrity and it depends on GSM services in order to provide secure communication. In addition, the only mechanism they employ to protect user's privacy is the ability for a given user to hide her position from everyone or from certain people. This is achieved by replying to an SMS asking her location either by providing it or not depending on the requestor identity. As discussed in Section 3.2, in order to protect user's location privacy, MILC combines different preferences for each user, real time notifications about location requests and the ability to disable a certain service. Work in (Safar et al. 2008) proposes a location based Personal Digital Assistant (PDA) application which is implemented using two system modules in tandem. The first module is actually an application which enables users to send their birth year and get their horoscope in return. The second module, namely "Find Friend", helps users to find which of their friends are moving close to them. Using the "Find Friend" feature users can enter their location data, like the city they are at the present moment, and then the application forwards it to the service provider. The result to the location query is a list of friends that roam in the same area. In (Safar et al. 2008) the authors construct and analyze a network of friends using an existing account from a social network (i.e., www.hi5.com). The privacy of users and the integrity of transmitted messages are not addressed by the authors. Also, location privacy is not considered at all and the end-users are not protected from possible eavesdroppers.

In (Kawada et al. 2005) the authors propose an application based on the Session Initiation Protocol (SIP) and employ some filters for selecting which personal data become public and how frequently in order to manage privacy issues. They use the XML format to transfer information between the communicating parties and event notifications that include three attributes in encrypted form i.e., an event attribute (talking, eating etc), an event ID and the partner's information. These event notifications are used in the synthesizing phase of their proposal in order to protect privacy, whenever a new user is added in a group conversation between other members. Each user receives different information from others due to user's buddy-list and publishing filters. But even if a user does not publish her presence but another one publish that she is doing "something" with "someone" there is some kind of leakage although the user's ID is not revealed. The U-Theme Park service was introduced by (Han et al. 2005) and consists of four services i.e., instant locator, BF, attraction information and a tour path recommendation service. The authors tried to combine several context based location services but with no respect to user's privacy. Only some kind of per user privacy password protects the BF service. This means that each user has a password and for someone else to locate her is required to possess the same password. But, by providing the user's alias and password someone can trace a user continuously and without any approval. MILC contains a

Draft

locator service as well but as described in Section 3.2 location information is sent after getting the consent of the end-user.

The Hubbub system described in (Isaacs et al. 2002) is a mobile IM that tries to provide awareness, opportunistic conversations and mobility through the use of acoustic characteristics. Privacy is protected by enabling the user to delete or block another user from being able to access her location information. These methods may lead to information leakage if they are not handled correctly, i.e., if the blocked user is able to be informed about the blockage. As described in Section 3.2, MILC tries to overcome such leakages by sending to the requestor a fuzzy message that the GPS of the peer may be switched off or no GPS is installed on the device. The authors in (Bønes et al. 2006) introduce an architecture namely MedIMob for providing a secure enterprise IM service targeting to healthcare environments. Because of the sensitivity of the transmitted information, the authors present a system which employs authentication and encryption. Specifically, hybrid encryption is used for end-to-end communication at the application level, combining symmetric and asymmetric encryption algorithms. More analytically, the message is encrypted using a one-time session key and the session key is encrypted with the recipient's public key and delivered with the message.

Protecting location privacy is a difficult task to achieve because information may leak at many levels of the protocol stack as discussed in (Long Wong et al. 2007). The authors focus on the physical layer, and for transmissions from mobile station to base station they propose the change of the transmission mode from omni-directional to a shaped beam. By doing so, an attacker needs to be within the coverage of the beam pattern and possess many resources in order to attack the system. But even in this case the proposed beam pattern reduces the chances of a successful attack. In location-based mobile services, the user prefers to be advised about places to visit without her privacy being exposed. The work in (Qi et al. 2004a) studies location privacy for mobile users when they roam to foreign cells. The authors utilise blind signatures to provide anonymous authentication. According to their study, administrator agents are in charge to preserve users' identity anonymity and perform verification using an authorized anonymous-ID as a digital token. But anonymous authentication is only a small part of the location privacy problem. However, this scheme does not provide unlinkability and provable security (Bellare 1997) as has been shown in (Ren and Lou 2007). The same authors proposed a mechanism, namely re-confusion protocol, which once again employs a blind signature scheme to generate an authorized anonymous ID replacing the real ID of a legitimate mobile user (Qi et al. 2004b). Their aim was to eliminate any associations between the authorized anonymous and real ID of a given user. But this protocol has been proved to be ineffective as discussed in (Liao et al. 2006). The authors think that Qi's et al. registration protocol may not revoke the linkability between the real and authorized anonymous ID of a user and at the same time certain flaws exist in the re-confusion protocol. The work in (Mannan and Van Oorschot 2004) offers a comprehensive survey on secure public IM. The authors discuss threats related to IM, and pay particular attention to those provoked by IM worms. Location based services legislation is addressed in the work by (Gow 2004). The author reports on several findings from a study about introducing initiatives to location-based services for public safety that took place in the United States, Canada, and Europe. In particular, he examines the legal and regulatory aspects in relation to the circumstances under which one can collect, use and disclose location information obtained from the use of mobile technologies.

Until now, only few proposals attempt to address the issues of security and privacy in spatial messaging (Deriaz 2008). The FoxyTag system proposed in (Deriaz and Seigneur 2006) uses an automatic computational trust engine. This engine must be location and time aware to automatically assess the trustworthiness of a spatial message. Also, the authors capitalise on their first work by describing a spatial messaging security framework based on the reputation of each spatial message per se (Deriaz and Seigneur 2007). This mechanism is somewhat analogous to the well known PGP web of trust (Zimmermann 1995). Overall, the main concern of both the aforementioned works is how the user privacy and the message authenticity will be preserved. In any case however, the way the reputation of a user is calculated strongly depends on what the others think about you, which in turn—due to several reasons—may not be always accurate.

The most prominent commercial applications similar to MILC are BuddyMob, IMEasy and GFindster. BuddyMob described in (BuddyMob 2009) is a mobile application that supports both chat and location-based services where users can be imported from all IM and social network platforms. BuddyMob users can be authenticated using their e-mail accounts and a pseudonym. Another similar application to BuddyMob is IMEasy (IMEasy 2008), which combines map services with IM. GFindster described in (GFindster 2008) is a location based chat application, which enables participants to acquire the geographical location of other users near them or globally. Each user can be identified by a pseudonym (nickname or alias) when using the application, however, her permanent identity could be revealed. All the commercial aforementioned applications do not support confidentiality, integrity and location privacy. More details for these applications and a comparison with MILC are given in Section 4.4.

Draft

The above discussion showed that security is not an actual priority for most existing implementations or proposals in the field. Moreover, the majority of them do not address the privacy of the end-users and when they do so they turn out to be ineffective. MILC on the other hand tries to conflate the provision of modern social networking mobile services with an acceptable level of security and truly privacy-preserving features.

3 System description

As mentioned earlier in this paper, MILC tries to combine chat and location-based services and at the same time provide communication confidentiality and user privacy. MILC follows a simple and lightweight client-server architecture where all communication passes through the server, which is supposed to be trusted. In the following sections we describe each MILC's component more analytically.

3.1 MILC server

MILC server is implemented in Java. By using threads the server handles client connections and forwards messages to other clients. The server has a pair of 1024 bit RSA asymmetric keys and a public key certificate, all issued by the Certification Authority (CA) of the University of the Aegean. The keys are used during the client authentication process, which will be explained in detail in Section 3.3. It is also assumed that all clients hold a copy of the server's public key in the form of a base-64 encoded X.509 certificate issued by the CA of the University. Note that a base-64 encoded certificate is very easy to manage and transfer to virtually every mobile device as it is in plain text. The way the client receives the certificate is out of the scope of this paper, thought it can be (pre)installed manually on the client application or received via e-mail.

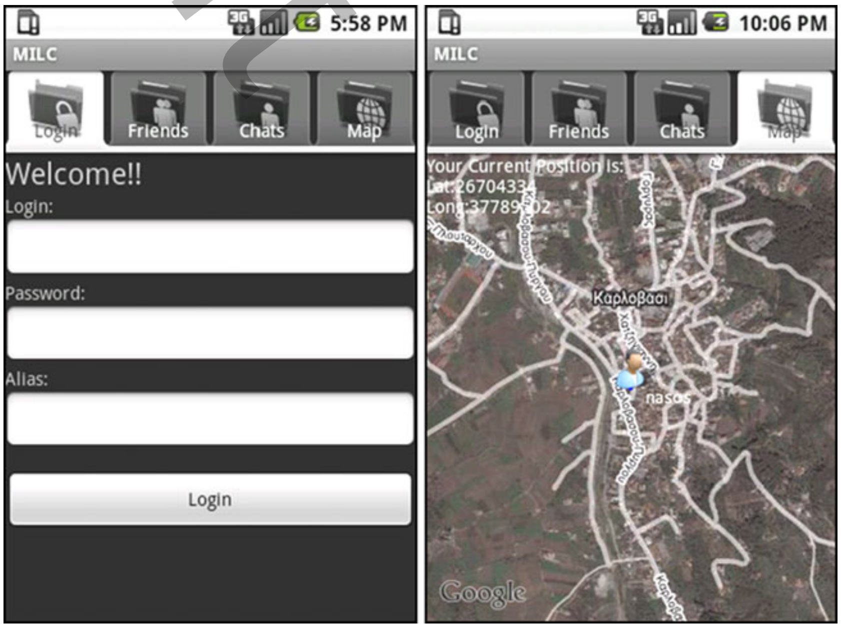
The server is also linked to the University's user database using Secure Lightweight Directory Access Protocol (LDAPS). This is a common method of securing LDAP (Sermersheim 2006) communication by using a Secure Sockets Layer (SSL) tunnel (Frier et al. 2006). By doing so, the authentication of the users will be carried out through credentials they already possess in order to access other (common) university services such as email. So, no registration is required for the students or the university staff. Of course, MILC can be easily embrace non university users by offering a registration page and a separate database.

3.2 MILC client

The MILC client prototype is implemented using Google's Android 2.0.1 SDK which is designated for developing applications for the latest release of Google's operation system. The interface of the client depicted in Fig. 1 consists of four tabs. The first one is the "Login" tab where a user provides her credentials in order to connect to the MILC server. Bear in mind that these credentials are the same as in any other service provided by the university. In

Draft

Fig. 1 The Login and map tabs



addition, there is the "Alias" field where users can provide a temporary per-session name in order to conceal their permanent identity. On the provision of the credentials a pair of 1024 bit RSA asymmetric keys is being created on the client side taking as input the credentials per se. This feature is implemented by using java.security.

After a client has been successfully authenticated with the server, every communication between both ends is secured with a symmetric session key that has been created by the server. This key is different for every client and for every session. Also, upon authentication the "Friends" tab comes in focus. This tab shows all users who are currently connected to the MILC server. Note that all users are shown by either their username or an alias, if one has been provided in the first place. Additionally, the tab "Chats" shows all the active conversations that a user has triggered with other users. For every conversation a brief history of the last 50 messages is kept in order not to overload the mobile device. Finally, the "Map" tab contains the corresponding Google's map which offers location information.

By selecting a user from the friends list displayed on the "Friends" tab, a chat screen appears. Prior to the activation of the chat, a symmetric session key for this conversation is created and installed on both clients. The key establishment procedure is also explained in detail in Section 3.3. From the same screen one can access location preferences as well. Here we can query any other user for providing us with her location and choose whether to send our location to others or not. There are four possible alternatives to choose from:

- Send the exact location every time another user asks,
- Send a relative location every time another user asks (meaning a location within a radius of 1 Km from our current location),
- Ask first and decide what to do and
- Do not send location.

When selecting the last option and in order to avoid information leakage, the requestor gets the general—and fuzzy—message that the GPS of the peer may be switched off or no GPS is installed on the device.

By using the aforementioned features, we can provide different options for different users and change our preferences on-the-fly. Moreover, there is a number of pre-settled Points of Interest (PoI) on the server that one can instantly retrieve in order to be informed about what facility is close to her current location. Of course, PoIs have to be constantly updated and populated to include more information. On top of everything else, MILC provides its users with the ability of SM. The location based operations are similar to the chatting ones and thus will be explained in Section 3.3 as well.

3.3 MILC protocol

This section describes in detail the three main procedures that are being carried out by the MILC's protocol. In the following we use the notation given in Table 1.

3.3.1 Authentication and key agreement

In this section we describe a simple lightweight authentication and key agreement protocol, which enables a user to securely authenticate herself with the MILC server. Our protocol utilizes both symmetric and asymmetric cryptographic operations. This procedure, which is depicted in Fig. 2, produces a 128 bit length symmetric key to serve as the session key ($sKey_i$). The authentication protocol is mutual. This means that it is used to authenticate the client to the server and vice versa. First off, the user provides her credentials i.e., {username, password} to the MILC client in order to connect to the service. Recall that the alias field is an optional value that will hide user's real username and ensure her pseudonymity. If no alias is provided, the username will be used instead. After the provision of the credentials an asymmetric key pair is being created on the client side using these credentials as a seed in order to ensure the singularity of the pair.

During the first step, the MILC client encrypts, using the server's public key (S_{pu}), the credentials and the alias. As already pointed out, the server's public key may be installed on each client. Then, the client sends the encrypted message along with the client's public key (C_{pu}) to the server and a thread between them starts. Upon receiving the first message, the server decrypts it, and via LDAPS, verifies that the requestor do exists in the database. If true, client authentication completes successfully and the server proceeds with the rest of the protocol. Then, the server creates a 128 bit AES session key and signs it, i.e. calculates its digest using the SHA-1 hash function and encrypts the digest using his private key (S_{pr}). After that, it sends towards the client the session key along with its digital signature encrypted with the client's public key that has been received previously at step one. The client, in turn, decrypts

Table 1 Notations used in MILC protocol

S_{pu}	Server's public key
S_{pr}	Server's private key
C_{pu}	Client's public Key
C_{pr}	Clients private Key
$sKey_i$	Session key for securing $client_i$–to-server communication
$cKey_{ip}$	Session key for securing $client_i$–to-$client_p$ communication
H()	SHA-1 hash
(x)y	Encryption of block x with key y

Draft

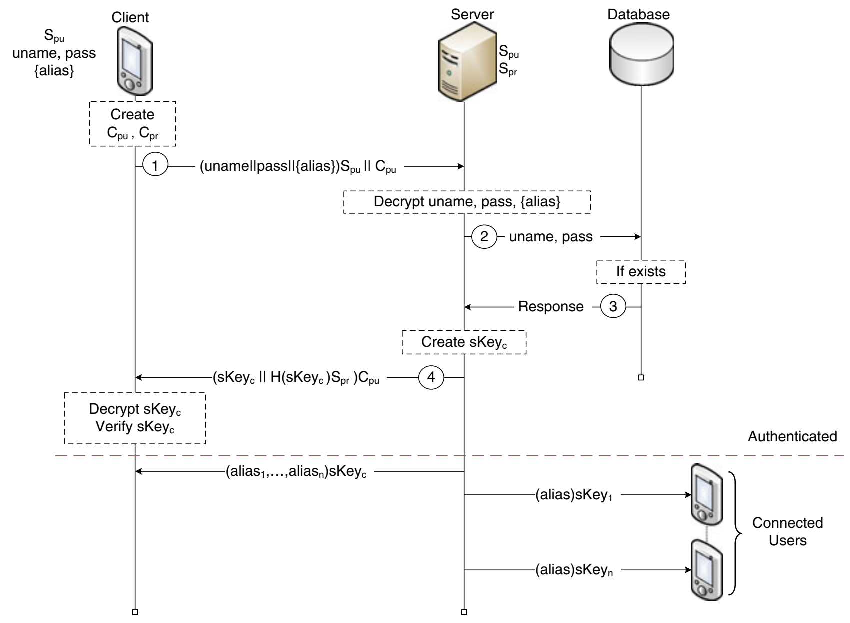


Fig. 2 Authentication and key agreement protocol

the message using its private key (C_{pr}) and obtains the session key and the server's signature. Then, it verifies the signature using the server's public key (S_{pu}) and if true the server has been authenticated as well. The mutual authentication between the client and server is completed in just two messages and from now on all communication is secured using the session key ($sKey_c$).

In addition, from this point on the user will be identified using the alias she provided. After authentication has successfully carried out, the server sends to the entrant a list of all connected users encrypted with the session key ($sKey_c$). Note that all users are identified by their alias or username as the case may be. The server also informs the online users about the new connection and sends to them the alias of the newcomer encapsulated with each client's session key.

3.3.2 Chat session establishment

Every time the user opens a chat session a new conversation is being initialized. Note that in the Fig. 3 below, the names initiator and peer are the corresponding client aliases and each client-to-server connection is secured using the corresponding symmetric session key ($sKey_i$ or $sKey_p$) been already installed on both ends.

Draft

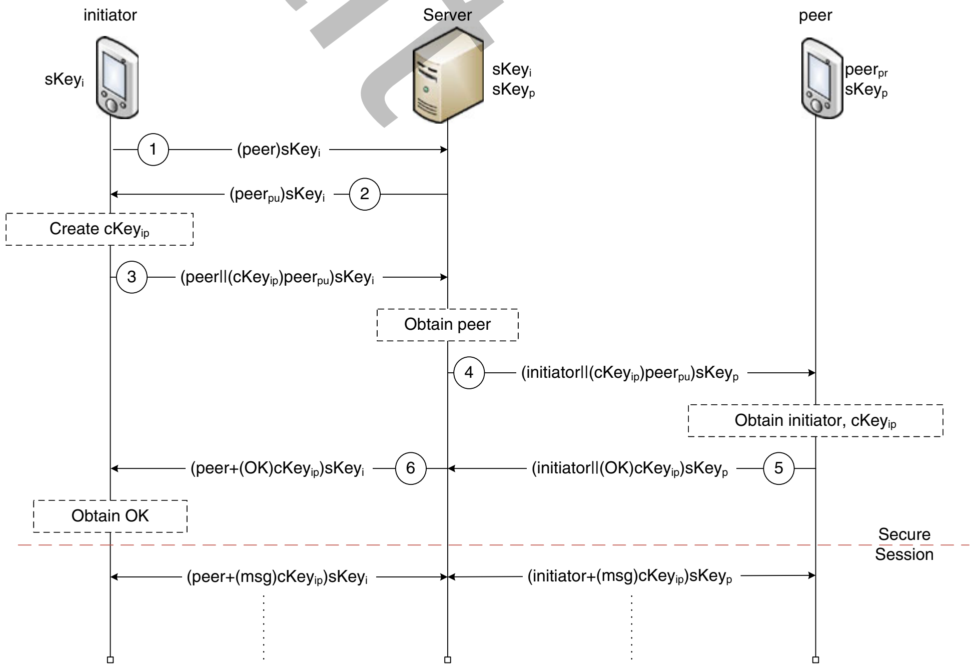

initiator
$sKey_i$
Server
$sKey_i$
$sKey_p$
peer
$peer_{pr}$
$sKey_p$
1 (peer)$sKey_i$
($peer_{pu}$)$sKey_i$ 2
Create $cKey_{ip}$
3 (peer||($cKey_{ip}$)$peer_{pu}$)$sKey_i$
Obtain peer
4 (initiator||($cKey_{ip}$)$peer_{pu}$)$sKey_p$
Obtain initiator, $cKey_{ip}$
(peer+(OK)$cKey_{ip}$)$sKey_i$ 6
(initiator||(OK)$cKey_{ip}$)$sKey_p$ 5
Obtain OK
Secure Session
(peer+(msg)$cKey_{ip}$)$sKey_i$
(initiator+(msg)$cKey_{ip}$)$sKey_p$

Fig. 3 Chat session establishment message flow

As already mentioned, during the authentication and key establishment phase, each client sends her public key to the server. These public keys are stored by the server in order to be used in chat initialization. Firstly, the chat initiator sends a message to the server asking for peer's public key. Actually, the initiator sends towards the server the alias of the peer encrypted with the session key ($sKey_i$). The server responds by sending back the requested public key ($peer_{pu}$). Note that it is less resource consuming to store all public keys on the server and upon request, send each key than to send all public keys at once and store them on the client.

Thereafter, the initiator creates another symmetric key ($cKey_{ip}$) that will be used to secure all subsequent chat messages between the initiator and peer and encapsulates it with the received peer's public key ($peer_{pu}$). Then, sends to the server a chat message including the alias of the user she wants to chat with (peer) and the encapsulated symmetric key. The server obtains the peer identity by decrypting the received chat message and forwards the encapsulated key to the peer after including the identity of chat initiator.

The peer obtains the identity (alias) of the initiator by decrypting the chat message with her session key ($sKey_p$), and also, retrieves the chat session key $cKey_{ip}$ by decrypting it with her own private key ($peer_{pr}$). Then she sends an "OK" message to the chat initiator encrypted with the chat session key ($cKey_{ip}$). Finally, the server forwards this message to the initiator of the chat session by replacing the initiator's alias with that of the peer. Note that all the above-mentioned messages between the server and the peer/initiator are encrypted using the corresponding session key ($sKey_i$ / $sKey_p$). Generally, the server is the forwarder of any message between the two entities, i.e., the initiator and peer. It receives a message carrying the alias of the peer and forwards it to her after including the alias of the initiator. In all cases however, the server is not able to access the chat session key ($cKey_{ip}$), as he does not know the clients' private key.

Note that the above procedure describes private chatting. MILC also supports multicasting and broadcasting. Multicast is the simultaneous postage of the same message to more than one recipients. In order to do that, the chat initialization process is being carried among a number of users, so that a group key is installed to each of them. Then, the message is encrypted using this group key and send to the server containing the aliases of the group. Finally, the server forwards the encrypted message to the members of group silently (i.e., without acknowledgement). On the other hand, broadcast is the postage of a message to all the connected users and thus, it does not include any key installation. The client encrypts a broadcast message using the session key ($sKey_i$) and sends it to the server who then forwards it to all the connected users using the corresponding session key.

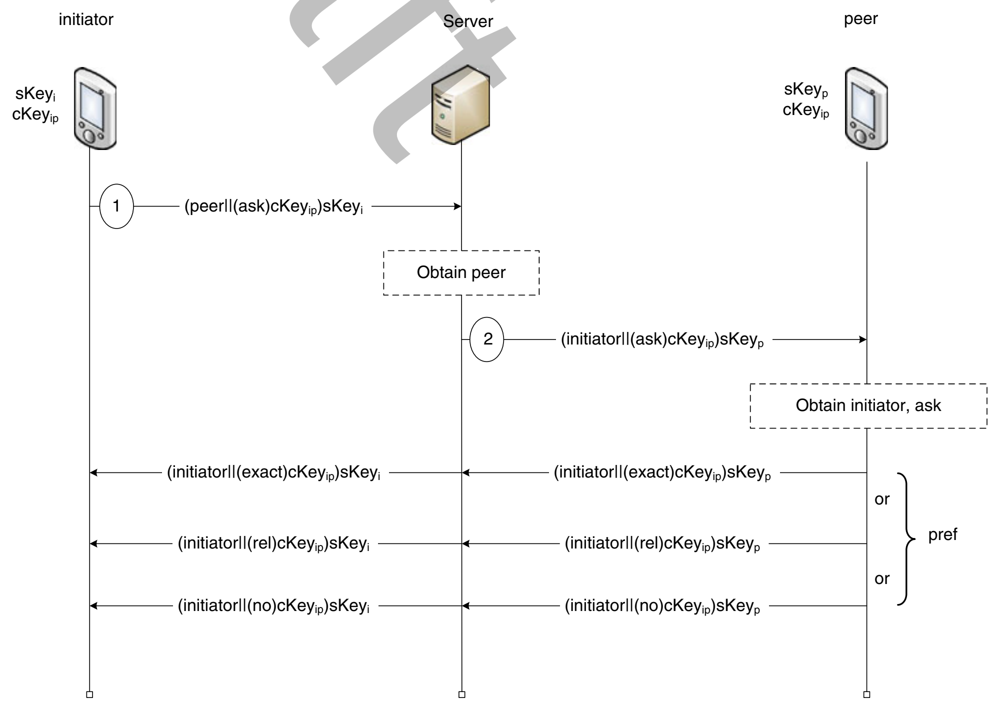
3.3.3 Location data acquisition

Location messages are a subcategory of chat messages. Before a user is able to ask for another user's location a chat session has to be active and thus, a chat session key ($cKey_{ip}$) must be installed on both clients. The necessary steps to complete a location request are depicted in Fig. 4.

First off, the initiator encrypts a location request message with the chat symmetric key ($cKey_{ip}$) and sends a message to the server containing the ID of the peer, i.e., its alias.

Draft

Fig. 4 Location request message flow


initiator
Server
peer
sKey$_i$ cKey$_{ip}$
sKey$_p$ cKey$_{ip}$
1
(peer||(ask)cKey$_{ip}$)sKey$_i$
Obtain peer
2
(initiator||(ask)cKey$_{ip}$)sKey$_p$
Obtain initiator, ask
(initiator||(exact)cKey$_{ip}$)sKey$_i$
(initiator||(exact)cKey$_{ip}$)sKey$_p$
(initiator||(rel)cKey$_{ip}$)sKey$_i$
(initiator||(rel)cKey$_{ip}$)sKey$_p$
(initiator||(no)cKey$_{ip}$)sKey$_i$
(initiator||(no)cKey$_{ip}$)sKey$_p$
or
or
pref

This message is encrypted with the session key ($sKey_i$). The server obtains the alias of the receiver (peer) and forwards another message to her including the alias of the sender (initiator). The peer in turn, obtains the alias of the sender by applying the session key ($sKey_p$) to the received message and retrieves the original location request message by decrypting the encapsulated message with the chat session key ($cKey_{ip}$).

The answer of the peer depends on the type of location information the user is willing to provide to the initiator. Recall from Section 3.2 that there are three possible answers (*exact, relative and no location*). Similarly to chatting session establishment, the server is unable to obtain the contents of neither the location request message nor the reply because it does not possess the chat session key ($cKey_{ip}$).

Another type of location request message a client can issue is that of PoIs. As mentioned in Section 3.2 by using such a request the client is able to retrieve a number of pre-settled PoIs in order to be informed about what facility is close to her current location. In this case, the peer sends a location message to the server containing her own alias and her relative position, i.e., a location within a radius of 1 Km from its current geographical position. When the server receives the message, he obtains the alias and the geographical coordinates by decrypting them with the session key ($sKey_p$). If the encrypted alias equals the alias of the sender the server knows that this is a PoI request and replies to the sender by sending her the list of PoIs encrypted with the session key ($sKey_p$). That list contains only the PoIs that are in a range of 10 Km from the requestor's current position. This scheme is not only effective (i.e., the returned PoIs are in the correct geographical area) but also preserves user's location privacy by only disclosing her relative location to the server.

Finally, there is the feature to broadcast or multicast a location request message like an ordinary chat message and thus, the procedure will not be further analyzed. In any way, only the user is responsible for the provision and accuracy of any location information she reveals to others. It is stressed however that such data is always send encrypted using either the session or group key. To avoid DoS incidents in case of simultaneous and continuous location requests by the same or different users, a time limit of 15 min for broadcast and 5 min for group/private location requests has been set. This time limit also provides a better privacy level as the user cannot be traced precisely even if she had agreed to provide her exact location upon a request.

3.3.4 Spatial messaging

Every MILC user, passing any physical point on the map, is able to leave a message for another user, a group of users or for everyone to pick up at a later time. When the corresponding user(s) pass near that point, the message would automatically appear to them after confirmation. This service is implemented using a MySQL transparent database so that even if the server is compromised, no information about the users or the spatial messages they have posted could be revealed. Users are able to activate / deactivate the SM service at any time. This is because this service may undermine the user's privacy, e.g., the server must track the user movement constantly or at specified time intervals. A participating user sends a spatial message to the server containing the coordinates of the geographical point, the list of users she wishes that message to be visible and a text message. The geographical points can be acquired either by getting the current location of the user using GPS or by simply selecting a point on map. The list of recipients is specified by employing their usernames, not the aliases. This is because the username is static and thus it is assured that the specified user will receive the message while the alias can change anytime. Then, the server, stores an SM record in the database using as primary key the triplet {coordinates of the given point, SM author, time in ms}.

If a user explicitly gives her consent to enable the service, her device sends location updates to the server every 2 min. The server in turn, queries the database for the existence of any spatial messages in a radius of 100 m. If true, it retrieves the message, checks if the user is in the recipients list and if so, forwards it to the corresponding device.

4 Evaluation

In this section we evaluate MILC in terms of performance, security, and privacy. Our aim is to demonstrate that MILC has a low level of computation times and network overhead and at the same time maintains a high level of security and privacy. In order to measure the performance of MILC we created a properly configured testbed and three separate scenarios. In the first scenario, we give a comparison between two versions (modes) of MILC; the standard (secure) one and a modified insecure version. As a metric we use the mean authentication time, i.e., the overall time required for a user to be authenticated by the MILC server. According to the second scenario, we calculate the mean time for chat session establishment in the secure version of MILC. This time includes all necessary actions to establish a symmetric key between two clients as already described in Fig. 3 (i.e., messages 1 to 6). Finally, the third scenario measures the time penalty imposed by the cryptographic operations, i.e., for the encryption and decryption of messages in the secure version of MILC. Section 4.2 focuses on various attacks that may undermine MILC availability and discusses how our system copes with such incidents. MILC privacy properties are discussed in Section 4.3. Finally, a comparison of security/privacy

Draft

characteristics of MILC with three other commercial mobile applications is given in Section 4.4.

4.1 Performance

The secure version of MILC has been already described in Section 3.3.1. The authentication procedure of the insecure version of MILC is similar to the secure but without the key creation phase and the cryptographic operations. That is, it includes only two steps; (a) the client sends his username and password to the server in cleartext, (b) the server sends back an acknowledgment message, whether the client has successfully been authenticated or not.

Figure 5 depicts the topology of our testbed composed by the MILC server and six clients placed in three different subnetworks. More specifically, the experimental network architecture comprises from the following elements:

- One server with an Intel Xeon CPU at 3.2 GHz and 3.8 GB of RAM, which plays the role of the MILC Server. As already pointed out, MILC application is implemented in java. Server's operating system is Fedora 11 with kernel version 2.6.30.9-102.fc11. x86_64. Our server connects to the network through a Broadcom NetXtreme Gigabit Ethernet card.
- One high-end laptop machine with an Intel Core 2 Duo P8700 at 2.53 GHz and 4 GB of 1,066 MHz DDR3 RAM, which connects to the network throughout the 56 Mbps AirPort wireless network card. The operating system of the machine is Mac OS X Leopard Snow. This PC is used as the MILC Emulator 3 in Fig. 5.
- One laptop machine with an Intel Core 2 Duo T7200 CPU at 2 GHz and 3.2 GB of RAM, which connects to the network through a 54 Mbps wireless network card. The operating system of this machine is Microsoft Windows 7. This PC is used as the MILC Emulator 1 in Fig. 5.
- One low-end laptop machine with an Intel Celeron Dual Core T3000 at 1.8 GHz with 2 GB of RAM, which connects to the network through a 56 Mbps wireless network card. The operating system of the machine is Windows Vista. This PC is used as the MILC Emulator 2 in Fig. 5.
- Three HTC Magic mobile devices which incorporate a Qualcomm CPU at 528 MHz and 288 MB of RAM. All these devices connect to the Internet through the incorporated IEEE 802.11 b/g wireless module.

Each subnetwork is located in different geographic area and employs two of the above clients, an HTC Magic mobile device and a laptop machine using the Google's Android Emulator with the 2.0.1 SDK. This will give us an indication of the degree the network distance, between a client and the server, affects the overall authentication time. Also, such setting allows us to measure possible differences in performance between real devices and emulator. The mean ping time for each subnetwork is given in Fig. 5, but

Draft

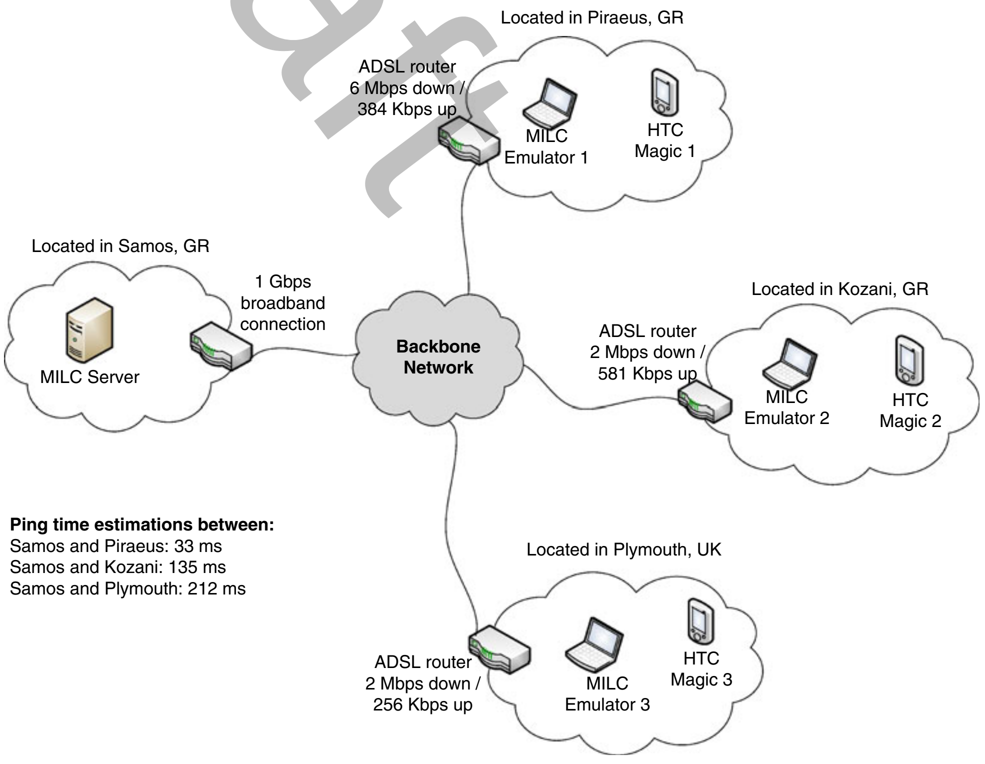


Fig. 5 Testbed network architecture

these values can only be considered as an indication. All clients connect to the Internet through a local 802.11b/g hot spot with each subnetwork having different connection speed. The subnetwork in Piraeus has a 6 Mbps Internet connection, while the subnetworks in Kozani and Plymouth connect to the Internet over a 2 Mbps ADSL connection. During all experiments the MILC server was providing services to at least 20 other clients to simulate a fair server overhead.

According to our first scenario we measured the average time required for a client to authenticate with the MILC server. The scenario begins when the client device starts creating a key pair and terminates the moment it verifies the received session key. For each client we have collected measurements from 50 runs performed during different times and dates. The clients one by one connect to the server and start to acquire services. Note, that the server is always kept busy serving other clients. This procedure was followed for both the secure and insecure version of MILC. Figure 6 depicts the mean time in ms required for each client to authenticate with the server for both MILC versions. The observation of Fig. 6 reveals that the overall authentication times of the secure version of MILC are significantly larger than the ones of the insecure scenario. For instance, the HTC client 1 using the insecure mode requires 265 ms to be authenticated with the server while the secure version takes almost 2,875 ms. This is translated to an increment of nearly 985%. Of course, this is due to the time the client requires to create the 1,024 bit key pair in the secure version, which in this case takes 2,546 ms. This means that the ≈88.5% of the total time is devoted to key creation and the rest i.e., 329 ms to authentication. It is stressed that the latter phase includes (a) the encryption/decryption procedures in both ends, (b) session key verification in both ends, and (c) network times (roundtrips). If we isolate the time required by only this phase in each configuration for the HTC client 1 we spot a 24% increment imposed by the secure mode (i.e., 265 vs. 329 ms). Naturally, this represents almost a negligible penalisation.

The MILC clients that run on emulators produce an unexpected augmentation of authentication delay in both modes. Specifically, for the secure and insecure mode we note an increment of ≈60.5% and ≈21% respectively. This may be because the emulator runs as a virtual device on the host machine and their interoperability is still on a low level. This explains the fact that although the connection times for the first two HTC clients differ by only 70 ms, which is mainly due to client–server subnetworks distance and general network conditions, for the corresponding emulator clients differs by 1,053 ms.

So, in order to have a better estimation we have included in Table 2 the minimum and maximum delays, the standard deviation of the taken measurements and the 95% confidence interval. These times correspond to the general case and take into consideration measurements gathered from all clients in each category. The observation of the table reveals that with a probability of 95% the overall connection time (i.e., key creation and authentication) for the HTC client using the secure mode is expected to span between 2.8 and 3.1 s which is of course acceptable by the end-user. Also standard deviation of all values is about 55% of the mean. This is actually a high value which is naturally affected by sporadic peak authentication times logged during the measurements. This is also confirmed by the fourth column of Table 2. Certainly, the main reason for this behavior is the volatile nature of the wireless connection itself.

The second scenario measures the average time required for an initiator to establish the chat session key with a peer (see Fig. 3). We examine three different cases:

Draft

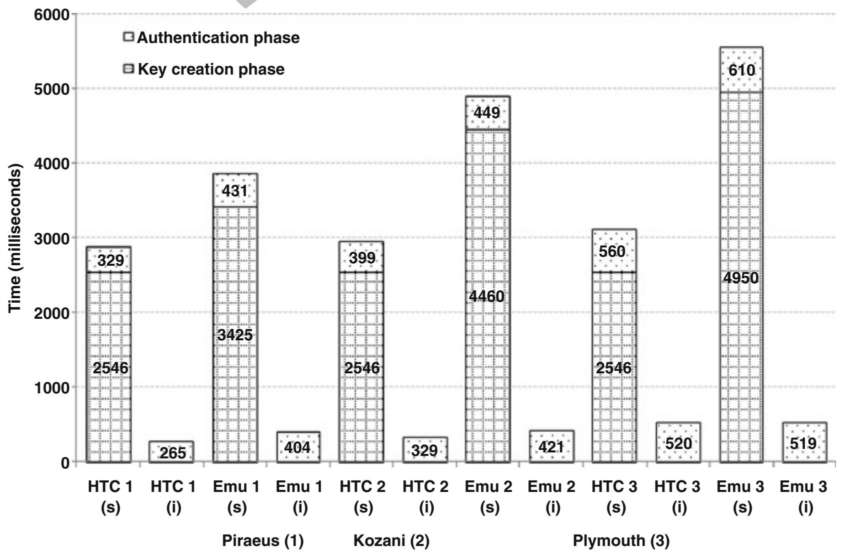


Fig. 6 Average authentication time per subnetwork for each client (s) = secure mode, (i) = insecure mode, Emu = emulator

Table 2 Statistical measurements in milliseconds for the first scenario

Configuration (mode)	Mean	Min	Max	Standard deviation	Confidence interval (95%)
Secure mode (HTC)	2975	1066	5254	1587	(2816, 3134)
Secure mode (Emulator)	4775	1778	10895	2726	(4501, 5049)
Insecure mode (HTC)	371	222	690	142	(351, 391)
Insecure mode (Emulator)	448	218	972	187	(421, 475)

- both the initiator and the peer are located in Piraeus
- the initiator is located in Piraeus while the peer in Kozani
- the initiator is located in Piraeus while the peer in Plymouth

For both ends we use the HTC device and for each pair we gathered measurements from 50 runs performed during different times and dates.

The observation of Table 3 shows that session key creation time is proportional to network distance. For example, the chat initialization procedure, when both clients are located in Piraeus (Piraeus—Piraeus pair), requires about 399 ms for the chat session key to be installed on both clients. On the other hand, the same procedure for the Piraeus—Kozani pair requires about 619 ms, i.e., an increment of ≈55%. Recalling from Fig. 5 that the ping time in the first case is 33 ms while in the second is 135 one can easily conclude that the difference in chat initialization times is mainly due to network condition. This difference is more apparent in the third case (Piraeus—Plymouth pair) where we note an increment of ≈100% and ≈30% from the first and second case respectively. In every case, this time is expected to be less than 1 s, which of course is tolerable by the end-user.

The last scenario tries to estimate the penalty imposed by cryptographic operations in the client side, i.e., message encryption and decryption. Once the authentication or chat initialization procedures have been completed, the corresponding 128 bit AES keys have been generated and installed to protect client-to-server or client-to-client connections respectively. The mean time for the HTC client to create the AES session key is 0.122 ms. Naturally, these times may vary depending on the hardware employed. We have created 6 test messages consisted of 10, 100, 300, 600, 800 and 1,000 characters respectively, and we gather encryption and decryption times from 20 runs per message, in order to calculate the mean time. Figure 7 depicts the derived results in ms. As expected, the more characters the message has the more time it takes to encrypt or decrypt it. Furthermore, we notice that the time required to encrypt a message is significantly greater than that of decrypting it. This can be explained due to the use of encoding and decoding Base64 functions. Specifically, Java's Base64 code for decoding large arrays is faster than encoding as explained in (UWYN 2007). Also, the encryption and decryption time always depends on the programming language employed (Java in our case) and on the number of applications running in client's background.

Table 3 Statistical measurements in milliseconds for the second scenario

Pair	Mean	Min	Max	Standard deviation	Confidence interval (95%)
Piraeus – Piraeus	399	339	541	70	(388,410)
Piraeus – Kozani	619	435	814	133	(598,640)
Piraeus – Plymouth	803	534	2319	500	(724,882)

Finally, Table 4 presents the resources in Kbytes, which comprise indicative values for a mobile device to run MILC. Due to MILC's dynamic nature is not possible to measure precisely the amount of RAM required because the more conversations we have the more memory MILC consumes. Upon initialization, MILC occupies ≈16 Mb of RAM while during a normal use occupies ≈19 Mb. The maximum value we have faced is ≈21 Mb which is after a whole day use with GPS enabled and more than 15 active conversations.

4.2 Security

When discussing the security level of MILC we must refer to and examine possible threats and attacks that the system is able to cope with. In this context, Table 5 summarizes all basic security requirements and the corresponding threats that can undermine the smooth operation of MILC. The MILC system provides mutual authentication, as already pointed out in Section 3.3.1. Thus, users who are not members of the MILC community are not able to access services, and at the same time, legitimate users can rest

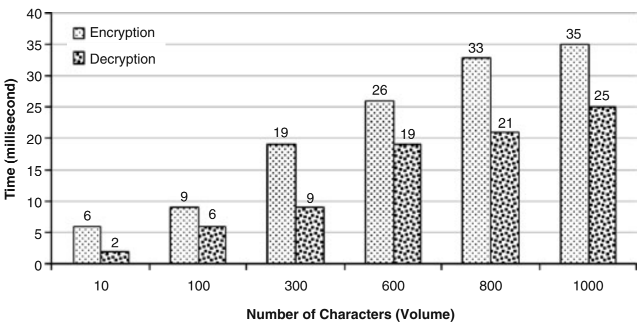


Fig. 7 Average encryption/decryption times for different payloads (HTC client)

Draft

Table 4 Memory recourses in Kbytes

Disk space for MILC application files (code, server's certificate etc)	236
Disk space for MILC's saved data	8
Total disk space for MILC application	244
Ram space required for MILC application	19,000

assure that communicate with the genuine server. MILC uses symmetric session keys for providing confidentiality, so eavesdropping or data leaking is considered very difficult, if not infeasible.

We try to balance wisely between the provision of security services and performance. Therefore, no extensive integrity mechanism is implemented as it would add unnecessary overhead to both the client and the server. Only during the authentication phase the session's key integrity is assured due to its triple importance. Firstly, this key enables the parties to become mutually authenticated, as it authenticates the server to the client (see Section 3.3.1). Secondly, it secures the corresponding client–server session by providing confidentiality of user's data and protection against eavesdropping and information leakage. And thirdly, as the authentication is mutual, the client can rest assure that all connected users are legitimate.

Another major security aspect that systems face and must cope with is DoS attacks. In MILC, DoS attacks split into two categories; attacks originated from "outsiders" or/and from "insiders". The first category includes DoS from non legitimate University members. Bear in mind that the MILC server is located inside the University network which is continuously available and under the protection of the University's firewall. Therefore, DoS and other disruption of service threats, caused from aggressors of this category, should be repelled by usual means and methods. The second category encompasses deliberate or accidental attacks from legitimate users. MILC is self-protected from this kind of attacks by providing time limits for location requests, as already pointed out in Section 3.3.3. Naturally, the MILC server, which is considered trusted, has access to user's username and temporary identity (alias) and is the only one who is able to link them. However, this is the only information the server can access. In fact, the server cannot even distinguish the type of a particular message it receives, i.e. if it corresponds to a chat or location transaction. Also, the server cannot access the actual contents of chatting data because all messages are encrypted using a per session symmetric key known to only the initiator and the peer. So, even in case the server is compromised the attacker is not able to eavesdrop on ongoing communications between the members of the MILC community or link an alias with a username as no log files are kept. Last but not least, MILC's default location privacy preferences are considered safe as well. As explained further on in Section 4.3 the user is always in absolute control of which specific personal information she wants to disclose to others.

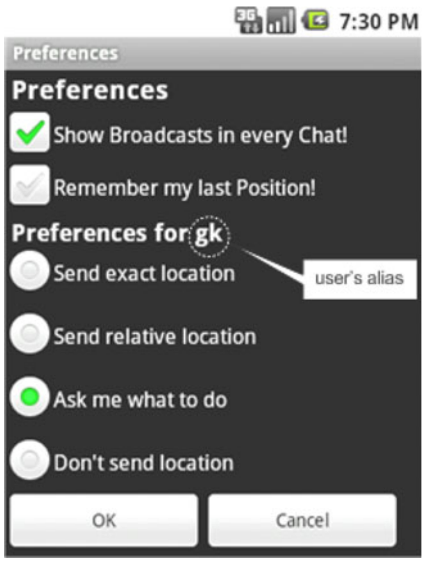
4.3 Privacy

MILC preserves end-user privacy by supporting pseudonymity and location privacy. This is realized by putting the user in control of her personal data. Firstly, pseudonymity can be provided on a per session basis. That is, each time the user connects to the MILC server is offered with the possibility to choose a different pseudonym for the current session. Also, location privacy depends every time on user selection. As already pointed out, MILC provides the service of asking the location of another user. Each time a user is requested to send her location an alert comes into focus. Therefore, the user is able to choose one of three alternative options: (a) to send her exact location, (b) to send a relative location, and (c) not to send her location. In case the user selects "relative location" a nearby location is sent (e.g., within a radius of 1 Km from the exact geographical position of the user). On the other hand, if the user selects not to send his location, a "GPS is disabled" or "no GPS is installed" message is send. This is to prevent information leakage. This means that the recipient of the message cannot be sure if the GPS is not working or the location did not send, on purpose, by the peer. Also, a user is able to set for any given user, different and per session settings, in order to automatically respond to a request. This situation is depicted in Fig. 8 which is the preferences screen of the MILC application.

Draft

Table 5 MILC's security requirements and associated threats

Security requirements					
Confidentiality	Integrity	Availability	Privacy	Pseudonymity	Authentication
Security threats					
Eavesdropping, Leaking of Data	Malicious modification of authentication and/or communication Data	Service Disruption (DoS)	Leaking of Geo-Location data, Leaking of User Identity, Insecure Default Preferences	Leaking of Users Identity	Unauthorized Access, Impersonation


7:30 PM

Preferences

Preferences

✓ Show Broadcasts in every Chat!

Remember my last Position!

Preferences for gk

Send exact location

Send relative location

● Ask me what to do

Don't send location

Fig. 8 MILC's preferences

As discussed in Section 3.2, there are the four possible responses to choose from, with the "*Ask me what to do*" being the default action for every new session. If any of the others is selected, then every time the user receives a location request from "gk" the system will reply automatically according to the specified value. Finally, if a user hangs up on a conversation, then automatically the disjointed user disappears from any active map. Overall, MILC provides a good and easily adjustable level of user privacy by giving the opportunity to the end-user to select the level of privacy she desires.

4.4 Comparison with related work

In this section we compare MILC with three other similar commercial applications; BuddyMob, GFrindster and IMEasy. Note that to the best of our knowledge, no other work in the literature combines instance messaging and location-based services into a unified system. The comparison is based on the following six basic criteria; mutual authentication, confidentiality, pseudonymity, resistance to DoS caused by insiders, integrity, and location privacy. As already pointed in Section 2, BuddyMob is a mobile application, where the participants (friends or "buddies") can be imported through all major IM and social network platforms and contacted directly through the application. A Geo-location service like the one described in (BuddyMob 2009) can be added on top of BuddyMob allowing participants to find out the location of any other joining member. In addition, any member is able to track her friend's geographical position by employing Google Maps. Also, the user receives location alerts when a friend moves close to her. However, location privacy in BuddyMob is only valid to protect the members of this service from guest users, thus all the registered members share their location information at any time, but guest users cannot access such information. Note that the term "guest" corresponds to users who access IM services like Facebook, MSN, AIM, GTalk, Jabber and Twitter through the BuddyMob application without having a BuddyMob account. In all IM systems users must be authenticated using their e-mail accounts and a pseudonym. Nevertheless, the real user identity can be easily exposed through her e-mail account. This is because, IM platforms are well-known not to use encryption and integrity mechanisms during authentication and exchange of user messages (Mannan and Van Oorschot 2004). This also means that all the IM platforms, are not able to confront eavesdropping and DoS attacks caused by insiders.

IMEasy is an instant messaging based application, than combines map capabilities with IM. A map is shown to all connected members interactively and synchronously. As already pointed out, IM platforms support authentication and pseudonymity but they do not offer any kind of confidentiality and integrity mechanism. Also, the map view can be saved as landmark. When one starts displaying a map to Alice, he is able to show her the landmarks shared with Bob two months ago. But Alice and Bob do not need to know each other. So, IMEasy does not support location privacy as well.

GFindster is a location-based chat application which enables participants to locate other users near them or globally and after that trigger a chat session with them. Private chat is supported as well. Each user can use a pseudonym in order to connect to the application. On the

Draft

Table 6 Application Comparison based on security requirements

Security requirements	BuddyMob	GFindster	IMEasy	MILC
User Authentication	Yes	Yes	Yes	Yes
Server Authentication	No	No	No	Yes
Confidentiality	No	No	No	Yes
Pseudonymity	Yes[a]	Yes[a]	Yes[a]	Yes
Resistance to DoS by insiders	No	No	No	Yes
Integrity	No	No	No	Yes[b]
Location privacy	Yes[c]	No	No	Yes

[a] The pseudonym and permanent user id can be associated

[b] Only during authentication

[c] Only for guest users

other hand, the Internet Relay Chat (IRC) used by the application is not safe because the IRC space is insecure with respect to confidentiality (Mannan and Van Oorschot 2004). The only protection IRC has, is that user's identity should be restricted to user's nickname. However, on systems like DALnet, where a permanent nickname is provided, the disclosure of user's identity is technically easier. Administrators have access to the network and are able to eavesdrop on all active conversations. To avoid such a situation many users employ a bouncer (proxy). But, even in this case, by opening a direct connection with another user, i.e., by sending a file, an eavesdropper can find her real IP just by monitoring his network status. So, he is able to associate an IP with different pseudonyms used from time to time, thus compromising the privacy of the end-user. On top of that, in case an attacker breaks into the IRC server log files, he is able to unveil the real identity of the user based on her IP address. Generally, it is well known that the IP address reveals information about the user's identity and this is a well documented issue in the literature. On the other hand, MILC avoids eavesdropping by using encryption, and as the messages are forwarded necessarily through the server, there can be no linkage between a given IP and one or more aliases. Furthermore, the absence of all kind of log files makes it even harder, if not infeasible, for anyone to trace user's activity and profile her in the mid or long term. Also, GFindster does not offer location privacy to its users and does not protect the confidentiality and the integrity of application data in transit.

Table 6 offers an aggregated, comparative view of all the discussed applications considering the abovementioned seven basic criteria. As shown in the table all the applications support user authentication and pseudonymity. MILC additionally provides mutual client–server authentication. Moreover, the pseudonym of a MILC user cannot be associated with her permanent identity in any way. On the contrary, this is not true for any IM or IRC platform. Excluding MILC, BuddyMob is the only one supporting location privacy, but this applies for guest users only.

5 Conclusions

Mobile applications are expected to mushroom over the next few years. This is driven by several strong factors like the growing interest in smart-phones and the involvement of Internet players into the mobile realm. This is further supported by modern networks' capabilities and developers capitalising on open platforms. For instance, the continuous success of the iPhone and the adoption of Google's Android operating system by mobile hardware vendors and service providers stimulate the penetration of smart-phones into the market and the demand for sophisticated mobile services. In this context, mobile social networking applications gain popularity and increase the volume of their users rapidly. However, so far, most of them have failed to deliver truly secure and privacy-preserving services to their users. Everyone would agree that anyone who participates in a virtual community needs to rest assure that any information she sends and receives remains confidential and that her private sphere is not violated without her consent.

In this paper we present the MILC system which is classified under the umbrella of mobile social networking applications. Specifically, MILC integrates in private or closed communities of scope individuals that participate in the community, mainly for educational reasons (students' communities, research groups etc.). MILC tries to address the aforementioned issues by (a) utilising both asymmetric and symmetric cryptography to provide a high level of security to its users, and (b) respecting end-user privacy by putting the user in control of what private information is revealed to other parties and under what circumstances. We provide a detailed description of the MILC prototype components, discussing their functionality and analyzing their aspects. We also demonstrate that MILC is lightweight in terms of service times. Also, we believe that our design can be used as a template for anyone interested in building, expanding and deploying a MILC-like system. As a statement of direction, we are currently working on enhancing MILC to support and further improve distance services offered to the academic community.

Draft

References

Askwith, B., Merabti, M., Shi, Q., & Whiteley, K. (1997). Achieving user privacy in mobile networks, in *Proc. 13th Annual Computer Security Applications Conference, ACSAC,* 108–116. doi:10.1109/CSAC.1997.646180.

Bellare, M. (1997). Practice-oriented provable-security, in *Proc. 1st International Workshop on Information Security* (ISW '97), 221–231.

Bønes, E., Hasvold, P., Henriksen, E., & Strandenæs, T. (2006). Risk analysis of information security in a mobile instant messaging and presence system for healthcare. *International Journal of Medical Informatics, 76*, 677–687. doi:10.1016/j.ijmedinf.2006.06.002.

BuddyMob (2009). BuddyMob. Retrieved from http://www.buddymob.com/. Accessed September 2009.

Chen, G., & Rahman, F. (2008). Analyzing privacy designs of mobile social networking applications, in *Proc. IEEE/IFIP International Conference on Embedded and Ubiquitous Computing, EUC '08, 2, 83–88.* doi:10.1109/EUC.2008.156.

Deriaz, M. (2008). The Uncertainty of the Truth, in Proc. 2008 Sixth Annual Conference on Privacy, Security and Trust, PST '08, 177–188.

Deriaz, M., & Seigneur, J. M. (2006). Trust and Security in Spatial Messaging: FoxyTag, the Speed Camera Case Study, in Proc. 2006 International Conference on Privacy, Security and Trust:

Bridge the Gap Between PST Technologies and Business services, PST '06. doi:10.1145/1501434.1501459.

Deriaz, M., & Seigneur, J. M. (2007). Towards trustworthy spatial messaging. *Electronic Notes in Theoretical Computer Science, ENTCS, 179*, 123–133. doi:10.1016/j.entcs.2006.08.036.

Frier, A., Karlton, P., & Kocher, P. (2006). The SSL 3.0 Protocol. Retrieved from http://home.netscape.com/eng/ ssl3/draft302.txt. Accessed December 2009.

GFindster (2008). GFindster Review. Retrieved from http://www.androidapps.com/t/gfindster. Accessed August 2009.

Gow, A. G. (2004). Pinpointing consent: location privacy, public safety, and mobile phones, in Proc. Conference on the Global and Local in Mobile Communications.

Han, T. D., Yoon, H. M., Jeong, S. H., & Kang, B. S. (2005). Implementation of personalized situation-aware service, in Proc. 1st International Workshop on Personalized Context Modeling and Management for UbiComp Applications, ubiPCMM '05.

IMEasy (2008). Introduction to Hi AIM. Retrieved from http://im-easy.com/. Accessed September 2009.

Isaacs, E., Walendowski, A., & Ranganathan, D. (2002). Hubbub: A sound-enhanced mobile instant messenger that supports awareness and opportunistic interactions, in Proc. SIGCHI conference on Human factors in Computing Systems: Changing our world, changing ourselves, SIGCHI '02, 179–186.

Kawada, M., Mimura, N., Morikawa, H., & Aoyama, T. (2005). A user-oriented presence synthesizing system for facilitating on-line communication, in Proc. 2005 Symposium on Applications and the Internet Workshops, SAINT '05 Workshops, 242–245. doi:10.1109/SAINTW.2005.1620021.

Liao, J., Qi, Y. H., Huang, P. W., Rong, M. T., & Li, S. H. (2006). Protection of mobile location privacy by using blind signature. *Journal of Zhejiang University – Science A, 7*(6), 984–989. doi:10.1631/jzus.2006.A0984.

Liu, K., Giannella, C., & Kargupta, H. (2006). An Attacker's View of Distance Preserving Maps for Privacy Preserving Data Mining, in Proc. 10th European Conference on Principles and Practice of Knowledge Discovery in Databases, *PKDD* '06. doi:10.1007/11871637_30.

Long Wong, F., Lin, M., Nagaraja, S., Wassell, I., & Stajano, F. (2007). Evaluation Framework of Location Privacy of Wireless Mobile Systems with Arbitrary Beam Pattern, in Proc. 5th Conference on Communications Networks and Services Research, CNSR '07, 157–165.

Mannan, M., & Van Oorschot, P. C. (2004). Secure public Instant Messaging: A survey, in Proc. 2nd Annual Conference on Privacy, Security and Trust, PST '04, 69–77.

Palazzi, C. E. (2004). Buddy-Finder: A Proposal for a Novel Entertainment Application for GSM, in Proc. 1st IEEE International Workshop on Networking Issues in Multimedia Entertainment, NIME '04, GLOBECOM 2004.

Qi, H., Wu, D., & Khosla, P. (2004a). The quest for personal control over mobile location privacy. *IEEE Communications Magazine, 42*(5), 130–136.

Qi, H., Wu, D., & Khosla, P. (2004b). A Mechanism for Personal Control over Mobile Location Privacy, in Proc. IEEE/ACM 1st International Workshop on Broadband Wireless Services and Applications, BroadWISE '04.

Quercia, D., Ellis, J., & Capra, L. (2010). Nurturing Social Networks Using Mobile Phones. *Journal of Pervasive Computing*, IEEE. PP(99). doi:10.1109/MPRV.2010.43.

Ren, K., & Lou, W. (2007). Privacy-enhanced, attack-resilient access control in pervasive computing environments with optional context authentication capability. *Mobile Networks and Applications, 12*(1), 79–92. doi:10.1007/s11036-006-0008-7.

Ruppel, P., Treu, G., Küpper, A., & Linnhoff-Popien, C. (2006). Anonymous user tracking for location-based community services. *Location and Context-Awareness, LNCS, 3987*, 116–133. doi:10.1007/11752967_9.

Safar, M., Sawwan, H., Taha, M., & Al-Fadhli, T. (2008). Virtual social networks online and mobile systems, in Proc. 1st International Conference on Applications of Digital Information and Web Technologies, ICADIWT'08, 119–126.

Sermersheim, J. (2006). Lightweight Directory Access Protocol (LDAP): The Protocol. Retrieved from http://www.rfc-editor.org/rfc/rfc4511.txt. Accessed December 2009.

UWYN (2007). Use what you need. Class Base64. Retrieved from http://rifers.org/docs/api/com/uwyn/rife/tools/Base64.html. Accessed 15 September 2009.

Zimmermann, P. (1995). *The official PGP user's guide*. Massachusetts: MIT.

Zhong, H., Bi, L., Feng, Z., & Li, N. (2008). Research on the design methods of mobile social network services, in Proc. International Conference on Information Management, Innovation Management and Industrial Engineering, ICIII '08, 2, 458–461. doi:10.1109/ICIII.2008.206.

Draft

Athanasios Loukas received the Diploma in Information and Communication Systems Engineering in 2008 and the MSc degree in Information and Communication Systems Security in 2010, both from Dept. of Information and Communication Systems Engineering, University of the Aegean. Currently, he is a PhD candidate at the Dept. of Information and Communication Systems Engineering of the University of the Aegean with research interests in Wireless Networks Security and Privacy, Mobile and Ubiquitous Systems Security and Mobile Services and Applications.

Dimitrios Damopoulos is currently a Ph.D candidate at the Dept. of Information and Communication Systems Engineering, University of the Aegean. He received an MSc in Information & Communication Systems Security from the Dept. of Information and Communication Systems Engineering, University of the Aegean, Greece. He also holds a B.Sc in Industrial Informatics from the Technological Educational Institute (TEI) of Kavala, Greece. His research interest includes Mobile Security, Mobile Device Intrusion Detection Systems, Mobile Applications and Services, Social Engineering.

Sofia A. Menesidou received the Diploma in Information and Communication Systems Engineering from the University of the Aegean (Greece) in 2008 and the MSc degree in Information and Communication Systems Security from the University of the Aegean in 2010. Currently, she is a PhD candidate at the Dept. of Electrical and Computer Engineering of the Democritus University of Thrace.

Maria E. Skarkala holds a diploma in Information and Communication Systems Engineering from the University of the Aegean, Greece (2008). In 2010 she completed her MsC in Information and Communication Systems Security from the same university. Currently, she is a PhD candidate at the University of the Aegean with research interests in privacy preserving data mining techniques and algorithms, information and communication systems security and mobile communications security and privacy.

Georgios Kambourakis received the Diploma in Applied Informatics from the Athens University of Economics and Business in 1993 and the Ph.D. in Information and Communication Systems Engineering from the Department of Information and Communications Systems Engineering of the University of Aegean. He also holds a M.Ed. from the Hellenic Open University. Currently, Dr. Kambourakis is a Lecturer at the Department of Information and Communication Systems Engineering of the University of the Aegean, Greece. His main research interests are in the fields of mobile and wireless networks security and privacy, VoIP security and mLearning. He has been involved in several national and EU funded R&D projects in the areas of Information and Communication Systems Security. He is a reviewer of several IEEE and other international journals and has served as a technical program committee member in numerous conferences.

Stefanos Gritzalis holds a BSc in Physics, an MSc in Electronic Automation, and a PhD in Information and Communications Security from the Dept. of Informatics and Telecommunications, University of Athens, Greece. Currently he is the Deputy Head of the Department of Information and Communication Systems Engineering, University of the Aegean, Greece and the Director of the Laboratory of Information and Communication Systems Security (Info-Sec-Lab). He has been involved in several national and EU funded R&D projects. His published scientific work includes 30 books or book chapters and more than 200 journal and international refereed conference and workshop papers. The focus of these publications is on Information and Communications Security and Privacy. His most highly cited papers have more than 800 citations (h-index = 15). He has acted as Guest Editor in 20 journal special issues, and has leaded more than 30 international conferences and workshops as General Chair or Program Commitee Chair. He has served on more than 200 Program Committees of international conferences and workshops. He is an Editor-in-Chief or Editor or Editorial Board member for 14 journals and a Reviewer for more than 40 journals. He has supervised 10 PhD dissertations. He was an elected Member of the Board (Secretary General, Treasurer) of the Greek Computer Society. His professional experience includes senior consulting and researcher positions in a number of private and public institutions. He is a Member of the ACM, the IEEE, and the IEEE Communications Society "Communications and Information Security Technical Committee".

Draft