International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:1, No:6, 2007

Dynamic Anonymity

Emin Islam Tatlı, Dirk Stegemann, and Stefan Lucks

***Abstract*—**Encryption protects communication partners from disclosure of their secret messages but cannot prevent traffic analysis and the leakage of information about "who communicates with whom". In the presence of collaborating adversaries, this linkability of actions can danger anonymity. However, reliably providing anonymity is crucial in many applications. Especially in context-aware mobile business, where mobile users equipped with PDAs request and receive services from service providers, providing anonymous communication is mission-critical and challenging at the same time. Firstly, the limited performance of mobile devices does not allow for heavy use of expensive public-key operations which are commonly used in anonymity protocols. Moreover, the demands for security depend on the application (e.g., mobile dating vs. pizza delivery service), but different users (e.g., a celebrity vs. a normal person) may even require different security levels for the same application. Considering both hardware limitations of mobile devices and different sensitivity of users, we propose an anonymity framework that is dynamically configurable according to user and application preferences. Our framework is based on Chaum's mix-net. We explain the proposed framework, its configuration parameters for the dynamic behavior and the algorithm to enforce dynamic anonymity.

***Keywords*—**Anonymity, context-awareness, mix-net, mobile business, policy management

I. Motivation

IN social life, people do not like to speak about their personal secrets in public. They do not want others to learn which illnesses they have, which books they buy and read, how much money they have in their bank accounts, etc. In the digital world, this scenario does not change very much. Users do not want others to learn which web pages they visit or to whom they send e-mail [17]. People prefer to stay anonymous whenever possible, i.e. they want to use resources or services without disclosing their real-world identities [9].

Pseudonyms provide a partial solution for anonymity. When communicating with a partner, a user introduces himself not with his real identity, but with a faked name called *pseudonym*. Pseudonyms are partial solutions, because even passive attackers who can sniff incoming and outgoing messages to/from the network nodes can link pseudonyms and reveal the real identities of users. Hence, unlinkability of user actions is required for providing anonymity [12].

Today, the most promising solutions for anonymity and unlinkability can be categorized into three [14]: *proxies* (e.g. Anonymizer [10]), *mix-net* [11] and *peer-to-peer networks* (e.g. Crowds [16] and Tarzan [13]). In the m-business framework that we will consider in further detail, mix-net based protocols are more suitable approaches than proxies and peer-to-peer networks. The anonymity level mix-net provides is higher than the anonymity level of proxies and in the m-business framework, direct communication between mobile users (*not peer-to-peer*) is not concerned.

Mix-net was first suggested by Chaum in 1981 for anonymous e-mail communication. Today, mix-net based solutions like ISDN-Mixes [15], Web Mixes [3], SMTP Remailers [7] etc. are widely deployed in various application scenarios.

However, integrating a mix-net-based anonymity solution into the m-business framework yields some additional challenges. Firstly, conventional mix-net protocols rely on computationally expensive public key operations, which are hard to execute on mobile devices with limited computational capacities.

Secondly, a mix-net that presents a fixed level of anonymity is undesirable since different applications may require different anonymity levels. As an example, a mobile dating application may require a higher anonymity level than an application for finding the nearest restaurant. Moreover, different users tend to have different sensitivity for security and may require different anonymity levels even for the same application. For example, a person with a high security sensitivity (e.g. a celebrity) still may require a high anonymity level for the finding the nearest restaurant service.

Varying sensitivities of users and applications demand a dynamic anonymity framework that allows users to specify the anonymity level for each application individually. In this paper, we propose such an anonymity framework and show its dynamic behavior based on the different configuration parameters.

This paper is organized is follows. Section II explains the m-business project and its framework principals in detail. The configuration parameters used for the dynamic anonymity behavior are discussed in Section III. The anonymity framework and its main components are described in Section IV, and finally Section V concludes the paper.

Manuscript received May 20, 2005. This work was supported in part by the Landesstiftung Baden-Württemberg and the Ministry of Science, Research and Arts of the state of Baden-Württemberg.

Emin Islam Tatlı is with the Computer Science Department, University of Mannheim, Mannheim, Germany (phone: 49-621-181-267; e-mail: tatli@th.informatik.uni-mannheim.de).

Dirk Stegemann is with the Computer Science Department, University of Mannheim, Mannheim, Germany (e-mail: dirk.stegemann@informatik.uni-mannnheim.de)

Stefan Lucks is with the Computer Science Department, University of Mannheim, Mannheim, Germany (e-mail: lucks@th.informatik.uni-mannnheim.de).

II. The M-Business Project

The presence of context-aware and especially location-aware m-business applications in our daily lives is steadily increasing. Using a PDA to order a pizza from the nearest restaurant that offers the pizza for less than a certain price is a typical example of a context- and location-aware mobile business service. It belongs to mobile business because it relies on a mobile PDA. It is context-aware because the maximum price query requires the context information *price* to exist in the system. Similarly, the nearest restaurant can only be determined in a location-aware framework. Besides restaurant-finding application, many more context-aware and location-aware m-business applications exist in the field. Locating kids [8] and people in emergency [4], locating moving objects (e.g. fleet management [1]), location-based chat and games [5], indoor and outdoor routing [2] are examples of already implemented m-business services. In the near future, the number and types of applications will increase similarly to the demands of mobile end-users. This introduces many new challenges, e.g. usability, adaptability and especially security that influence user acceptance and determine the success of m-business services.

The m-business research project [6] that we base our analysis on aims to build a generic framework for different types of context-aware and location-aware m-business applications. In this framework, *mobile users*, *broker* and *service providers* are the main principals. Mobile users equipped with PDAs are interested in getting free or paid services from service providers. The broker is the central principal in the framework. It registers the services of the service providers within its repository. Upon request for a particular service, it returns a list with descriptions of services matching the query to mobile clients. The clients then apply to one of the service providers in their list to get the service.

Considering all security challenges in the m-business framework [18], anonymity is the most challenging and the most critical one. As motivated above, a fixed anonymity level for *all* applications and for *all* mobile users in the framework is unacceptable. Therefore, our anonymity framework takes into consideration the different anonymity needs of users and applications as well as the performance limitations of mobile devices.

III. Dynamic Anonymity

Our anonymity framework is based on mix-nets. A *mix* is a computer between the sender and the receiver of a message. Instead of sending the message directly to the receiver, the sender submits the message to the mix-net, which routes the message through a number of mixes to the receiver. This strategy makes it much more difficult for a traffic-analyzing adversary to identify sender and receiver of an intercepted message. More precisely, the sender encrypts the message with the public key of the receiver and additionally with the public keys of all mixes through which the message is routed. Upon getting an encrypted message, each mix decrypts the message with its private key and forwards it to the next hop, which is either another mix or the final receiver.

To achieve dynamic anonymity, a number of configuration parameters are supported by the proposed framework. The values of the parameters can be updated dynamically and different anonymity levels for different applications running on the same device are thus provided.

A. Configuration Parameters

We define six different types of parameters: *encryption type*, *mix number*, *path picker*, *message threshold*, *dummy message and time delay*. These parameters affect both the anonymity level of applications and the system performance.

1) Encryption Type

This parameter can take the values of *asymmetric* or *symmetric*. If it is set to *asymmetric*, the sender encrypts the message with the public key of each mix. Otherwise, the sender initially generates a secret key for each mix on the message route and distributes the generated shared key to the mix in a secure way, e.g. by encrypting the shared key with the public key of the mix. After this key handshake process, the sender starts sending messages anonymously by encrypting them with the shared keys of the mixes. This parameter is especially useful for the m-business framework since some mobile devices may not be capable of performing public key operations at all or at least not at sufficient speed.

2) Mix Number

This parameter specifies the maximum number of mixes in the mix-net through which the messages are exchanged. If it is set to zero, sender and receiver connect directly to each other, i.e. the anonymity component is disabled. The higher the mix number is, the higher the anonymity level is, but the network latency also increases with the number of mixes.

3) Path Picker

This parameter can take three different values: *sender*, *firstmix* and *random*. The value *sender* implies that the sender itself defines the message route and encrypts the message for each mix accordingly. *firstmix* implies that the sender only picks the first mix on the path and encrypts the message for this mix. The first mix randomly chooses the rest of the path and encrypts the message for each mix accordingly. This option is very throughput-efficient because the sender needs to encrypt the message only for the first mix. But from a security point of view, it has the drawback that the first mix gets to know the sender and also the final receiver of the message. The third value *random* is similar as the *firstmix* option. The sender randomly chooses a mix and sends the message to this mix. But this time, the first mix does not decide on the entire rest of the path. With equal probability, it either chooses another mix and forwards the message to it, or it sends the message directly to the final receiver. The benefit of this option is that traffic analysis is more complicated (*due to randomness*) than in the *firstmix* option, but it also requires each mix to know the final receiver.

4) Message Threshold

Upon getting a message, a mix can either forward the

message immediately or keep it in its outgoing message pool until a certain number of messages exist in the pool. When this threshold is reached, the mix sends all the messages in the pool to their next hops in random order. The threshold value for the maximum number of messages is specified by the parameter *message threshold*. Increasing the threshold value complicates traffic analysis and increases the anonymity level.

5) Time Delay

This parameter has a similar functionality with *message threshold* and specifies how long (in minutes) the messages will be kept in the outgoing pools of mixes.

6) Dummy Message

Increasing the number of messages sent among network principals complicates the traffic analysis. If the parameter *dummy message* is set to *send,* the mixes periodically send forged messages to other mixes.

B. Anonymity Policy

In order to enable dynamic changes of anonymity parameters, different anonymity configurations called *polices* can be created for individual applications. Before an application starts to communicate, its anonymity policy is retrieved from the repository and the behavior encoded in the policy is enforced accordingly. Fig. 1 illustrates a sample anonymity policy.

```
<policies>
 <policy id="1" belongsto="app_1">
   <configuration>
     <encryptionType>symmetric</encryptionType>
     <mixNumber>3<mixNumber>
     <pathPicker>sender</pathPicker>
     <messageThreshold>5</messageThreshold>
     <timeDelay>10</timeDelay>
     <dummyMessage>send</dummyMessage>
   </configuration>
 </policy>
</policies>
```

Fig. 1 Sample Anonymity Policy

C. Templates for Anonymity Policy

On the other hand, it is not very practical to expect that one defines a separate policy for each application. Hence, we define some default configurations called *templates*, which can be used readily by applications. Fig. 2 illustrates two samples of policy templates. The templates can be referenced from a policy (see Fig. 3). Thus, applications which do not have any particularly defined policies can rely on pre-defined configurations that implement various anonymity levels.

```
<templates>
 <template id="1" anonymityLevel="high">
   <configuration>
      <encryptionType>asymmetric</encryptionType>
      <mixNumber>10<mixNumber>
      <pathPicker>sender</pathPicker>
      <messageThreshold>50</messageThreshold>
      <timeDelay>60</timeDelay>
      <dummyMessage>send</dummyMessage>
   </configuration>
 </template></templates>
```

Fig. 2 Templates for Anonymity Policy

```
<policies>
  <policy id="1" belongsto="app_1" anLevel="low"/>
  <policy id="2" belongsto="app_2" anLevel="high"/>
</policies>
```

Fig. 3 Referencing a Template within a Policy

IV. The Framework

The anonymity framework consists of three main components: Application Manager, Anonymity Manager and Policy Manager. Fig. 4 illustrates the architecture of the framework integrated within a mobile client.

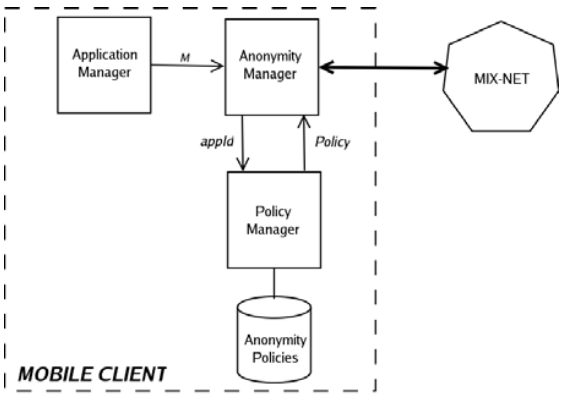

Fig. 4 Architecture of Dynamic Anonymity Framework

A. Application Manager

When an application running on the m-business application framework wants to send a message (*m*) over the network, firstly the Application Manager is notified for this transmission. The Application Manager creates another message (*M*) and forwards it to the Anonymity Manager to transmit it over the network. The message *M* consists of four different fields:

M= {*appID, m, receiver, anonymityLevel*}

appID is the unique id of the application that wants to send the message m. This field is used to retrieve the policy of the application from the repository. *m* is the message of the application to be sent and relevant only to the final receiver, i.e. service providers in our case. *receiver* specifies the final destination of the message. *anonymityLevel* is an optional field. If there is no pre-defined policy for the application, then this field can be used to specify the anonymity level required by this application.

B. Anonymity Manager

Upon getting the message *M*, the Anonymity Manager retrieves the anonymityLevel value if it is set, otherwise the *appId* value. This value is forwarded to the Policy Manager and gets the policy (*anonymity configuration*) back. Afterwards, the Anonymity Manager runs the algorithm (see Section IV-B.1) over the policy and decides how to interact with the mix-net and proceed with sending the message *m*.

1) The Algorithm

Three out of six policy parameters within are important for the anonymity algorithm of mobile clients. The *message threshold*, *time delay* and *dummy message* parameters only affect the behavior of mixes, but not mobile clients. Therefore, these 3 parameters are only forwarded to the mix-net. The algorithm process based on *encryption type*, *mix number* and *path picker* is illustrated in Fig. 5.

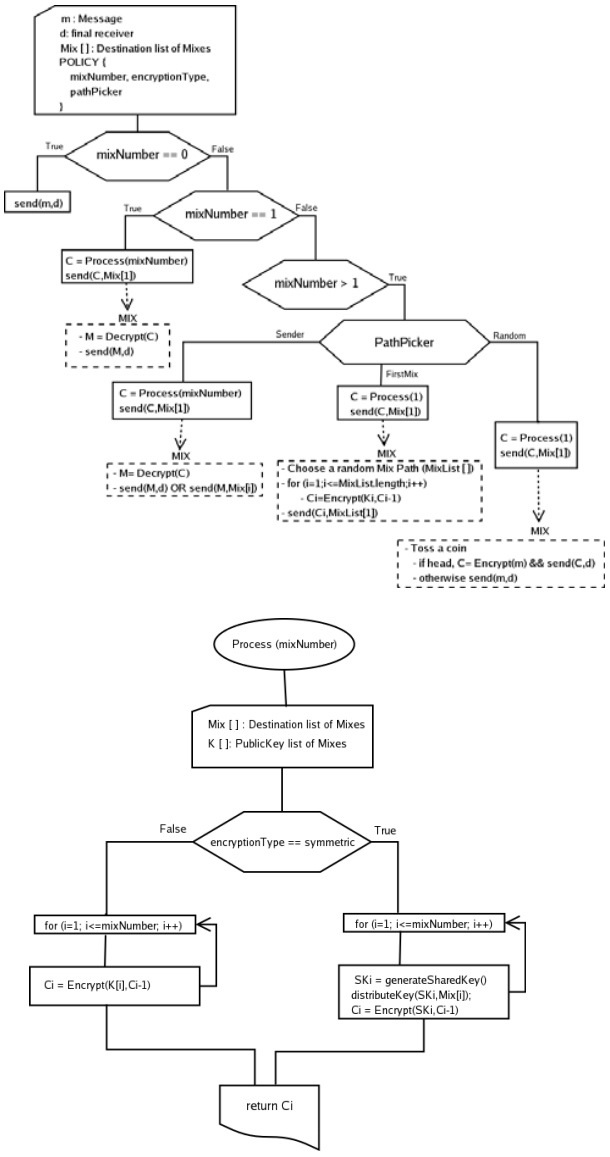

Fig. 5 The Algorithm over Policy

C. Policy Manager

The *Policy Manager* is responsible for storing and retrieving anonymity policies. It searches the anonymity policy of an application represented by an appId value forwarded by the *Anonymity Manager*. In case the Anonymity Manager forwards the anonymityLevel value, the Policy Manager retrieves the relevant policy from the template policies.

V. Conclusion

In the m-business framework which aims at building a generic infrastructure to support different types of context-aware services, one is faced with the challenge of anonymity of mobile users. Mix-net is the most promising solution for this application framework. But its requirement for time-consuming public key operations has turned out to be a bottleneck for mobile devices with limited computational capabilities. Additionally, fixed-level anonymity for *all* users and *all* applications is not an optimal solution since different users generally have different anonymity sensitivities, even for the same application. Taking into consideration the performance drawbacks of mix-nets, different sensitivities of users and applications, we have proposed a mix-net based anonymity framework that supports dynamic configurations for individual applications. The dynamic behavior of the proposed framework depends on six parameters, namely *encryption type* (i.e. symmetric or asymmetric), the *number of mixes* in the mix-net, *path picker* (i.e. sender, firstmix or random), the *threshold* and *time delay* values for the messages within the outgoing pools of mixes and *dummy message* specifying whether mixes would send blank messages among themselves.

References

[1] Fleet management.URL: http://www.fleetonline.ch.
[2] Indoor & outdoor routing.URL: http://www.falk.de.
[3] Jap: Anonymity and privacy tool for internet. URL: http://anon.inf.tu-dresden.de.
[4] Locating people in emergency.URL: http://www.sintrade.ch.
[5] Location-based chat and games.URL: http://www.vodafone.de.
[6] The mobile business research group.URL: http://www.m-business.uni-mannheim.de.
[7] Smtp remailers.URL:http://www.freehaven.net/related-comm.html.
[8] Tracking of kids.URL: http://www.trackyourkid.de.
[9] ISO99 IS 15408.URL: http://www.commoncriteriaportal.org, 1999.
[10] The Anonymizer. URL: http://anonymizer.com.
[11] David L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM*, 24(2):84–90, 1981.
[12] Andreas Pfitzmann et al. Anonymity, unobservability, and pseudonymity: A proposal for terminology, July 2000.
[13] Michael J. Freedman and Robert Morris. Tarzan: A peer-to-peer anonymizing network layer. In *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS 2002)*, Washington, DC, November 2002.
[14] Stefan Köpsell, Hannes Federrath, and Marit Hansen. Erfahrungen mit dem Betrieb eines Anonymisierungsdienstes. *Datenschutz und Datensicherheit*, 27(3), 2003.
[15] Andreas Pfitzmann, Birgit Pfitzmann, and Michael Waidner. ISDN-mixes: Untraceable communication with very small bandwidth overhead. In *Proceedings of the GI/ITG Conference on Communication in Distributed Systems*, pages 451–463, February 1991.
[16] Michael Reiter and Aviel Rubin. Crowds: Anonymity for web transactions. *ACM Transactions on Information and System Security*, 1(1), June 1998.
[17] Bruce Schneier. *Secrets and Lies*, chapter 5. Security Needs, pages 63–67. Wiley and Sons, 2001.
[18] Emin Islam Tatlı, Dirk Stegemann, and Stefan Lucks: Security challenges in location-aware mobile business, In *Proceedings of the 2nd International Workshop on Mobile Commerce and Services*, München, 2005. IEEE Computer Society.