Anonymity in a Mobile Computing Environment

N. Asokan
Department of Computer Science
University of Waterloo
Waterloo, Ont. N2L 3G1, Canada
nasokan@uwaterloo.ca

Abstract

In a mobile computing environment, it is desirable to protect information about the movements and activities of mobile entities from onlookers. Solutions to this problem of providing anonymity have to be developed with the constraints of mobile computing environments in mind. In this paper, it is argued that this issue merits investigation. A brief survey of the nature of anonymity provided in proposed or existing mobile computing environments is presented. It is argued further that achieving limited but practical anonymity using standard cryptographic techniques is feasible. Example solutions are presented.

1 Introduction to the Problem

In a distributed system where some entities (i.e. users and computing devices) are mobile, it is often desirable to keep information about their movements, locations and activities confidential. The modalities of providing such *anonymity* depend on the underlying security infrastructure of the system. Typically, this infrastructure is built using a shared key cryptosystem (SKCS), a public key cryptosystem (PKCS) or a hybrid system.

Anonymity has two aspects [4]. *Anonymity of location* deals with keeping an entity's movements and whereabouts confidential. *Anonymity of data origin/destination* deals with keeping an entity's activities confidential. The latter seeks to prevent an onlooker from associating the entity with messages sent to/from it or with the sessions in which it is a participant. Within the context of this paper, anonymity as a goal does not imply unrestrained anonymity; it implies the protection of relevant information from *unintended* parties. For example, I assume that a server may demand that a potential client reveal its true identity and perhaps provide adequate authentication as a prerequisite to the use of services.

In traditional computing environments, with fixed entities and wireline networks, anonymity of location has not been considered a significant issue. In a mobile computing environment, it assumes greater importance. It is likely to be an important security problem to solve before mobile computing gains wide acceptance. Anonymity is especially desirable when an entity travels outside of its 'home domain.'[1] In spite of that, it has not been considered a basic requirement in designing protocols for mobile computing environments (e.g. the IETF mobile-IP effort [12] has not yet addressed this issue). Like every other aspect of security, provision of anonymity is tightly coupled with the architecture of the rest of the system. For instance, providing anonymity of network layer identity will be of little use if the routing protocol uses re-direction: an onlooker who has access to a router elsewhere on the system can determine the whereabouts of a mobile entity by trying to send a message to it. Therefore, the problem of providing anonymity should be considered during the design stage of building mobile computing environments.

There are philosophical motivations for providing anonymity as well. An entity that chooses to protect its location and activities from unintended third parties must have the mechanisms available to do so even if openness and full disclosure is a common policy. Besides, the principle of least information is a good rule to live by in designing security systems.

The rest of this document describes some proposed solutions to the anonymity problem and makes suggestions for improving them. The basic theme behind the suggestions is that effective anonymity can be achieved by making a *limited disclosure* of information.

[1] A *domain* consists of entities that fall within a common administrative control.

2 Solutions

Notation

X_K - A message X encrypted/sealed by a key K.
K_{AB} - A secret key shared between parties A, B.
K_A, K_A^{-1} - Public and private key pair of A.
N_A - A *nonce* created by A.
M - Mobile entity.
R - Remote domain M is visiting.
H - M's home domain.
D - M's current domain. (H or some R)

2.1 Shared Key Cryptosystems

In a distributed system with a security infrastructure based on a SKCS, an entity (the 'client') authenticates itself to another entity (the 'server') by demonstrating the possession of a secret key shared between them. To do this, the client has to first announce its (claimed) identity in the clear. This makes it hard to provide anonymity of location.

An intuitive solution is to use the notion of *nicknames*. Molva et. al. [11] suggest that a mobile entity could use a *traveling alias* when it travels in foreign domains. Only the mobile entity and its home domain will know the mapping between the traveling alias M_t and the real identity M. They go on to suggest that M_t be changed "at regular intervals; perhaps as often as passwords or PINs." However, in the absence of a protocol to change traveling aliases over an insecure channel, entities can change traveling aliases only when they return to their home domain. Such traveling aliases with fairly long lifetimes may be problematic in certain circumstances. For example, as mentioned before, the mobile entity may be called upon to reveal (and even prove) its real identity M before being granted certain services in foreign domains. With time, the set of entities that know the mapping between M_t and M will grow. Further, an onlooker will be able to use a long lived alias M_t to link the various activities of M for the period during which the alias remains in effect.

An alternative is to use short-lived aliases [8]. Aliases are changed by mutual agreement between the mobile entity and its home domain server. This implies that the mobile and the home domain stay synchronized. If synchronization is lost, a sub-protocol is necessary for re-synchronization. Achieving re-synchronization securely is a challenge. In [8] the mobile entity may be required to send its identity in the clear in order to achieve resynchronization. This, as observed in the same document, constitutes a "breach in the provision of the service."

Loss of synchronization is likely to be rare. In addition, the home domain is likely to have significant computing resources at its disposal. Therefore, in the event of loss of synchronization, I suggest that it may be reasonable to expect the mobile M to simply send a message to H encrypted with K_{HM}. H can then search through its database of shared secret keys until it finds one that can extract the message. If the number of entities in H's jurisdiction renders this an expensive operation, then I suggest that H can divide the principals into groups of manageable sizes, each with a unique group identifier. When synchronization is lost, M can send an encrypted message along with its group identifier. This way, a reduced level of anonymity can be provided by making a *limited disclosure* of information.

Preserving anonymity of data origin is similar to preserving anonymity of location. Anonymity of data destination is somewhat different when the destination is a mobile entity. Carlsen's solution [4] suggests essentially (see below for a clarification) the following: a domain server D that wants to send a message to a mobile M currently present in its domain, will multicast a message $\{M, K_s\}_{K_{DM}}$ to all the mobile entities currently present in the domain. Each mobile will attempt to decrypt every such multicast message. Only M will be able to recover the message and the session key K_s which can be used to secure the subsequent session. Carlsen argues that in a portable (voice) communication system, such a computing load on mobile entities is reasonable assuming 3 calls per hour. The arguments may not be applicable in a data communication system where the frequency of messages to a mobile entity can be potentially much higher. Carlsen's model actually consists of various *ports* (or base stations in more common parlance). Each port serves a number of mobile entities that happen to be inside its region. Thus, it is a port which attempts to send a message to a mobile M without revealing M's identity to an onlooker. The description here is conceptually similar to Carlsen's solution.

2.2 Public Key Cryptosystems

Public key cryptosystems can provide location and data origin anonymity in a natural way. The mobile M can use the domain server D's public key K_D to encrypt messages to it. Within an administrative domain, each M can be required to know the public key K_H of its home domain server. However, when M wanders into a different domain R, it cannot be expected to know the public key, K_R. It has to identify itself first so that R can determine how to *mutually* authenticate M. The decrypt-on-the-fly solution by Carlsen [4] to provide anonymity of data destination (as described in the previous section) will be prohibitively expensive in the case of a PKCS.

Some researchers [7][9] have discouraged the use of a PKCS to build authentication protocols for environments with mobile entities. The rationale for this recommendation is that the computational complexity of known PKCSs are beyond the limited resources available to mobile entities. Even if this argument is valid for a specific environment, it is likely to be a temporary limitation. What is more relevant is the *asymmetry* in available resources between mobile entities and their stationary counterparts. Beller et. al. observe [2] that it is feasible to design practical authentication mechanisms using PKCSs keeping this asymmetry in mind.

2.3 A Hybrid Solution

Complete anonymity may be hard to achieve. But it might also not be an absolute requirement. The earlier notion of *limited disclosure* can again be used to achieve practical anonymity. For instance, it might be acceptable to reveal that the visiting mobile entity is from domain H (or a suitably large super-domain of H), without revealing its exact identity. A hybrid protocol can provide such limited anonymity without being too complicated. In the rest of this section I describe an example protocol for authenticating a mobile entity using the above notion of limited disclosure.

Consider the following model: the system is partitioned into domains. The authentication server in domain H shares a secret key with every entity in the domain. H's public key is known to every entity in its domain. Each pair of domains H and R have (or using an inter-domain key distribution protocol, can dynamically arrange to have) a shared key K_{HR}.[2] The remote server R will provide services to the visiting mobile M only if *(a)* M's true identity is revealed to it and *(b)* the authenticity of the claimed identity is adequately demonstrated. Finally, all network links are considered to be insecure channels in which an intruder can observe, modify or destroy data. The goal is to achieve satisfactory mutual authentication between M and R while minimizing the set of entities that can infer the presence of M in R by watching the traffic.

The authentication protocol works as follows:

When a mobile M shows up in a remote domain R, it begins the registration process by identifying its home domain H and presenting its credentials encrypted with the public key of H. I assume that M can compute the key K_{MR} which is to be used for the mutual authentication. I elaborate on the computation of K_{MR} below. The first message is sent from the M to R as follows:

M1: $M \Rightarrow R : Cred_{MR},\ H,\ \{M, R, N_M\}_{K_{MR}}$

Where,

$$Cred_{MR} \ = \ \{M, R, H, \{T_M\}_{K_{MH}}\}_{K_H}$$

T_M is either a time stamp (which will imply the need for loosely synchronized clocks) or some indicator of freshness (e.g. a nonce that was handed to M the last time it authenticated successfully to H). In this description, I will assume that T_M is the current timestamp. R cannot verify the credentials. It can however find and mutually authenticate H and pass the credentials to it. The second message is from R to H as follows:

M2: $R \Rightarrow H : \{R, Cred_{MR}, N_R\}_{K_{HR}}$, R

H can verify the veracity and timeliness of the message from M. If the verification succeeds, H has to send the necessary information back to complete mutual authentication between M and R. The exact nature of this information depends on the assumptions made. For example, if T_M is a nonce and not a timestamp, H has to generate a new nonce to be sent back to M so that it can be used in the next protocol run. H also needs to inform R (and perhaps M) of the key K_{MR} that is to be used to secure the channel between M and R. H can pick a random key K_{MR} to be used by M and R. Or K_{MR} can be derived from other information that is unique to a given protocol run. For example, K_{MR} can be a one-way hash function $h(K_{MH}, R, T_M)$. In this case, M already possesses enough information to compute K_{MR} when it initiates the protocol. I use the latter. This is why I stated that M can compute K_{MR} and use it to encrypt the second part of the message $M1$. The third message is from H to R as follows:

M3: $H \Rightarrow R : Tick_{HR}$

Where,

$$Tick_{HR} = \{R, H, M, K_{MR}, N_R\}_{K_{HR}}$$

R can now know the true identity of M. It can also extract the key K_{MR} using which it can extract

[2] Designing a scalable key distribution mechanism is another security problem that has been made more urgent by the advent of mobile computing. While there have been proposals to solve this problem by imposing a hierarchy of trust, such a hierarchy is not acceptable in all situations. This remains an open issue.

the second part of message $M1$. It can use these to complete the mutual authentication with M as follows:

M4: $R \Rightarrow M : \{N_R^{'}, N_M, M, R\}_{K_{MR}}$

M can use the already computed K_{MR} to extract this message. It can verify that this response is timely by checking for the presence of its nonce N_M inside. It can then respond to the challenge by demonstrating to R that it was able to extract the nonce $N_R^{'}$.

M5: $M \Rightarrow R : \{M, R, N_R^{'}\}_{K_{MR}}$

Several variations to this example protocol are possible. For example, in message $M5$, R can allocate a temporary identifier to M as in [11] to be used in subsequent messages from and to M while it is in R. The same K_{MR} is expected to be used for mutual authentication between M and R for subsequent connection attempts for the entire duration of M's stay in R. But R or M can force a re-authentication involving H at any time, should they feel the need to do so.

This protocol provides limited anonymity since an onlooker can only know that an entity from H visited R but cannot determine the exact identity of the entity. It is scalable in that it can provide anonymity to an arbitrary level of granularity: instead of specifying H, M could have identified some parent domain $H^{'}$ of H and used $K_{H^{'}}$ to encode its identity relative to $H^{'}$. While $H^{'}$ shares no secret with M and therefore cannot verify the authenticity directly, it can form a trusted path to H and pass along the credentials. This protocol also takes the asymmetry of computing power into account: the mobile entities transfer the burden of verifying authenticity to their home domains. Even if the inter-domain key distribution protocol was based on a PKCS, the expensive task of verifying a chain of certificates provided by R will fall on H and not on M. There is only one public key operation (encryption using H's public key) that M has to perform. In general, the public key operations of a PKCS are less expensive than the private key operations. M has to store a small, finite number of keys in its permanent storage(e.g. K_{MH} and K_H). It also needs to store K_{MR} (or enough information to re-compute K_{MR}) for the duration of its stay in R.

A correctness proof of this protocol using the BAN logic [3] appears in [1]. However, the proof only establishes that the authentication protocol is correct. In future work, I hope to investigate how the analysis techniques can be used to draw formal conclusions about the preservation of anonymity.

3 Related Work

A referee pointed me to the work done by David Chaum. Chaum and others have done extensive work on unconditionally anonymous protocols to implement, among other things, the digital equivalent of cash. The basic theme of security (for providing and/or using services and products) without identification is expounded by Chaum in [6]. By requiring that the security of a transaction not be dependent on establishing the identities of participants to one another, it is possible to design protocols that achieve complete unconditional anonymity. Unconditional anonymity also opens the possibility of "perfect crime" (See [13], page 123, for a reference). Chaum uses the phrase "limited anonymity" in [5] to denote the anonymity provided by a special set of protocols. These protocols, like the unconditional anonymity protocols, preserve the anonymity of participants from onlookers and from each other. However, if necessary (for example, by a court warrant), the anonymity of a transaction can be undone at a later time.

In this paper, I assume that the legitimate parties involved in a transaction can demand to know the real identities of the other parties. The concern here is to protect the identities of certain parties involved in the transaction from other onlookers who are not legitimate parties to the transaction. By "limited anonymity" I mean that some information about the identity of a participant is leaked to an eavesdropping third party.

The ideas presented here are intuitive. Similar ideas have been used in different contexts. In [10], Maxemchuk et. al. describe various protocols for providing anonymity using a building block called "Double Locked Box Protocol." Using this protocol, an entity A can send a message to another entity B via an intermediary I. A starts by sealing the identity of B in a box that only B's computer can open. This box is placed inside another box, along with the identity of B's computer and sealed so that only the intermediary I can open the outer box. The box is then handed to A's computer. A's computer cannot see what is inside but knows that it needs to be passed to I. I can get no information about the sender or recipient except the identities of their computers. B's computer, which receives the inner box from I knows nothing about the sender. As is evident, the basic idea is that by disclosing a limited amount of information in the clear, significant anonymity can be obtained.

4 Conclusion

Anonymity has not been a major issue in traditional distributed systems like the Internet. But the advent of mobile computing has provided strong motivations to address this problem. Providing complete anonymity is often hard, infeasible or undesirable. This document has made suggestions that can help provide limited but practical anonymity by using *limited disclosure* of information. Provision of anonymity, and security issues in general, must be taken into account while mobile computing environments are being designed.

5 Acknowledgements

I would like to thank Jay Black, James Godfrey, Anil Goel and Leena Tirkkonen for their careful reading of earlier versions of this paper and for their valuable comments and suggestions. Robyn Landers helped improve the quality of presentation.

References

[1] N. Asokan. Security Issues in Mobile Computing. Unpublished manuscript, October 1994.

[2] Michael Beller et al. Privacy and Authentication on a Portable Communications System. *IEEE JSAC*, 11(6):821–829, August 1993.

[3] Michael Burrows et al. A Logic of Authentication. Technical Report 39, Digital Systems Research Center, February 1989. Revised February, 1990; Appendix(The Scope of a Logic of Authentication) May, 1994.

[4] Ulf Carlsen. Optimal Privacy and Authentication on a Portable Communication System. *Operating Systems Review*, 28(3):16–23, July 1994.

[5] David Chaum. Untraceable Electronic Mail, Return Addresses and Digital Pseudonyms. *CACM*, 24(2):84–88, February 1981.

[6] David Chaum. Security Without Identification: Transaction Systems to Make Big Brother Obsolete. *CACM*, 28(10):1030–1044, October 1985.

[7] ETSI. DECT Common Interface: Part 7 - Security Issues. ETSI Standard ETS 300 175.7, European Telecommunications Standards Institute, October 1992.

[8] ETSI. GSM - Security Related Network Functions. ETSI Standard GSM 03.20, European Telecommunications Standards Institute, October 1993.

[9] Y. Frankel et al. Fraud Prevention and Availability. Manuscript available in ftp://software.watson.ibm.com/pub/security/cdpd, October 1994.

[10] N. F. Maxemchuck and S. Low. The Use of Communications Networks to Increase Personal Privacy. Available from ftp://ftp.research.att.com/dist/anoncc, August 1994.

[11] Refik Molva et al. Authentication of Mobile Users. *IEEE Network*, 8(2):26–34, March/April 1994.

[12] C. Perkins. IP Mobility Support. Internet Draft 07, IETF Network Working Group, October 1994.

[13] Bruce Schneier. *Applied Cryptography*. John Wiley & Sons, first edition, 1994.