Increasing Privacy in Mobile Communication Systems using Cryptographically Protected Objects*

Uwe G. Wilhelm
Laboratoire de Systèmes d'Exploitation
Ecole Polytechnique Fédérale de Lausanne, CH-1015 Lausanne
e-mail: wilhelm@lse.epfl.ch

Abstract

Confidential information contained in the transactional data created in mobile communication systems is not very well protected. We present an approach to protect entire objects against manipulation and disclosure, which allows to retain some control over information handed to a different entity or can be used to prevent reverse engineering of an object to obtain its original source. The approach is subsequently used in the context of mobile communication systems to protect the current location of a mobile user against unauthorized access.

1 Introduction

Communication has always had a considerable impact on our society. This observation pertains especially to systems for mobile communication, such as GSM (Global System for Mobile Communications), due to their omnipresence and ever increasing success. The fact that everyone is reachable at any time gives this form of communication a new quality that did not exist in the past.

The need for protecting the *communication contents* (payload data) in telecommunication systems against manipulation and disclosure (integrity and confidentiality protection) is well established and constitutes essentially a solved problem [15]. The protection of *transactional data* (such as who is communicating when, how often, and from where with whom) that contains equally sensitive information, is much less understood. This data could be used to construct a complete trace of the movements of a user carrying a mobile phone in standby-mode or a list of the people that user called on a given day. There are many examples where this data might contain information that users might rather have protected.

In existing telecommunication architectures, there are no means available to protect this transactional data, since it is essential for the operation of these systems (accounting, billing, or locating users to establish a communication channel). Thus, many of the employees of a telecommunication provider have potentially immediate access to the transactional data that is available in the telecommunication system. The only protection that is currently provided for this data are laws that prohibit its use for anything other than the intended purpose. Even though we generally assume that a telecommunication provider (such as a national phone company) is trustworthy and will adhere to these laws, we can not necessarily assume the same for each of its employees. This concern for the possibility of *malicious insiders* is founded on some recent publications that describe the vulnerability of systems towards their proper employees [12, 14].

*Research supported by a grant from the EPFL ("Privacy" project).

The fact that laws provide threats to deter most people from abusing the data that is available to them, might not be enough to prevent data abuse in the case when the benefits for the offending person outweigh the threats. This problem is further magnified by the fact that data abuse is inherently hard to detect and that it is very difficult to prove that it happened in the first place. We assume that a malicious insider has legitimate access to all the data that is available within a given telecommunication system. Therefore, this attacker model subsumes an external attacker who gains access to the system or an operational error that makes the data of the system accidentally available to outsiders.

In this paper we will present a new approach for protecting transactional data in telecommunication systems from unauthorized access, which is based on the *cryptographically protected objects* (CryPO) protocol (see Section 4).

The remainder of the paper is organized as follows. First, in Section 2, we will give a more precise definition of the problem concerning location information in mobile communication systems. We will also identify a more general problem that we will resolve first. In Section 3 we introduce some basic definitions and notations that we need for the description of the protocol. Next, in Section 4, we introduce the CryPO protocol in detail, which solves the more general problem. Then Section 5 describes the application of the general solution to mobile communication systems and discusses some results. Finally, Section 6 concludes the paper with a summary of the main contributions.

2 The Problem

The basic problem that we are concerned with is that once some information is disclosed to another entity, the original owner of the information looses all control over this information. The receiver can use it without any constraints, duplicate it, or even disclose it to a third party. Sending the data in encrypted form or with an attached signature does not prevent anything of the above.

This loss of control is inevitable if the goal of the interaction is that the receiver uses the information for some specific purpose. However, there are possible interactions, where the information is disclosed to the receiver just in case it is needed at a later time. The receiver will store it in an easily accessible place, but most of the time the information is not actually needed and simply discarded or overwritten with a more recent information.

We will first rephrase this problem for the protection of location information[1] in a mobile communication system and then describe a more general form of the problem that is concerned with protecting the code and data of an object against manipulation and disclosure.

2.1 The Protection of Privacy

We want to protect the location information in mobile communication systems, such as GSM (Global System for Mobile Communications)[2].

This system divides the entire coverage area of the mobile communication system into small compartments called *cells*, in order to make better use of the scarce bandwidth available for the radio link between a mobile terminal and a base station (which serves as intermediary to the wired network). These cells could be as small as a single room within a building or as large as a few kilometers in an unobstructed area.

For a user, say Alice, to remain reachable (available to receive incoming calls), she has to register the cell in which she currently resides with a well known location register (LR); in GSM this is called Home Location Register (HLR). If another person, say Bob, tries to reach Alice, he (or rather the underlying mobile communication system) will first contact Alice's LR, which

[1] We are particularly interested in the location information and will limit our discussion to this issue. However, the methods we propose for protecting it can easily be applied to other confidential data.

[2] GSM deals primarily with terminal mobility. Similar problems exists in the Intelligent Network service Universal Personal Telecommunications (UPT) that deals with personal mobility or the Universal Mobile Telecommunications System (UMTS) that deals with both forms of mobility. For reasons of brevity we will limit our discussion to terminal mobility.

is identified in her personal phone number, and obtain the cell in which she is registered. Then Alice will be contacted (*paged*) within this cell and informed about the communication request from Bob. If she accepts the call, the communication with Bob is established.

In the described system, which uses simplified concepts from GSM, the provider of the mobile communication system has complete access to the data stored in the LR and can easily detect in which cell Alice is currently located. However, for the operation of the system it is not necessary that the telecommunication provider knows Alice's location at all times, but it is sufficient to know her location when she has to be paged for an incoming call. Since it is impossible to foresee when this will occur, Alice has to send continuous updates on her current location, which will be stored in the LR and subsequently overwritten with more recent information whenever she moves to a new cell. This information could be used by the provider to create a continuous trace of Alice's movements (a location profile).

The location information is confidential data that should not be available to anyone without a compelling reason (such as law enforcement personnel with an order issued by a judge). Currently this data is solely protected by legal means, which make it unlawful to use it for anything other than the intended purpose (i.e., locating a user for an incoming call). Since this is not sufficient in the presence of malicious insiders, we want to add stronger protection based on technical means that allows only authorized access to this confidential information.

An approach to solve this problem has been proposed in [9]. It is based on a fixed terminal within the network (which we will call *HPC* for Home PC) that is under Alice's exclusive control. She sends continuous location updates to her HPC that are not accessible to any other entity. In the case of an incoming call, the telecommunication provider can query her HPC to obtain her current location. There are many possibilities how the HPC can react to such a query. The simplest would be to always disclose the current location of the user. This would not really protect the location information, but if the HPC keeps a log of all the queries, the user can verify that his location was only disclosed for genuine communication requests[3]. On the other extreme, the HPC could act as reachability manager [4] by requesting an authentication from each caller and accepting only communication requests from a set of users defined by the owner of the HPC (other callers can be deviated to a voice mail system).

Since Alice trusts her HPC to never disclose her current location without proper authorization, the goal of protecting the access to confidential information is achieved. Nevertheless, the described approach has some disadvantages:

- unreliability: the HPC is a computer under the administration of the user, with all the advantages and problems this situation entails. For many users it could be a major task to keep the HPC running.
- inefficiency: the HPC is outside the domain of the telecommunication provider and, thus, probably connected to the core of the system via a relatively slow link. Since the HPC can not be migrated within the network, this could be a major problem if the user moves to a different network for an extended period of time.
- vulnerability to:
  - attacks: the HPC is not particularly protected against physical attacks within the home of the user. A determined attacker could probably gain access to it and obtain the confidential data stored in the HPC. Moreover, as the HPC is connected to a communication system, it might also be possible to attack it without physical access.
  - traffic flow analysis: since all the traffic passing through the HPC belongs with high probability to its owner, the approach is quite vulnerable to traffic flow analysis. In [9] this problem is countered with the introduction of MIXES (see [3]), which allows to hide communication connections.

[3]The HPC can detect systematic attacks, such as continuous probing that could be launched to obtain a trace of the user's location.

- standardization: the approach described above is very different to those taken in most telecommunication systems. Telecommunication providers would probably be very reluctant to adopt it, primarily for reasons of reliability and efficiency.

An analysis of the approach reveals that the main idea is to encapsulate the confidential data in an object that provides access to the stored information only under the constraints of strict access control. The approach is valid if the object resides on a trusted environment that protects it from any access not supported by its interface. Is this not guaranteed, an attacker could obtain a copy of the object and analyse its contents[4]. We can therefore map the above problem to the following more general problem of protecting an entire object against manipulation and disclosure.

2.2 The Protection of Objects

We consider a *mobile object*[5] O, identified with its unique name O_{name}, that consists of code and data, which can be sent via a communication channel to a different host. There it can subsequently be executed on a virtual machine, which means that the object is platform independent.

A provider who sends an object to a receiver currently has no guarantee whatsoever concerning the confidentiality of the code or the data that comprise the object. If the communicated information constitutes a considerable economic value (e.g., proprietary algorithms) or contains confidential data, there is a potential problem. The receiver of the object could reverse engineer the code to obtain the original object source and reuse all or parts of the illicitly obtained code or he could analyse the object in order to obtain the confidential data.

This is not a new problem, yet it is perceived as a major threat in the context of *Java*, which seems to be particularly prone to reverse engineering. In the following Section 4, we will introduce the *cryptographically protected objects* (CryPO) protocol that can be used to overcome this problem. A rather similar approach for a different environment is described by Herzberg and Pinter in [6]. Their concern is for protecting software sold through regular distribution channels against piracy. The resulting protocols are quite similar, but the technical development has caused rather drastic changes to the computing environment so that a reevaluation of these ideas and an adaption to a more current context seems necessary.

3 Preliminaries

In order to describe our system, we need to introduce some important concepts from cryptography and tamper resistance.

3.1 Cryptography and Notation

A detailed description of cryptography and the corresponding notations is not within the scope of this presentation. Several books [2, 11] deal extensively with these questions. We will only introduce some basic notations needed for the description of the CryPO protocol.

The protocol uses public key cryptography (such as RSA [10]), in which a *principal* P has a pair of keys (K_P, K_P^{-1}) where K_P is P's public key that is known to everyone and K_P^{-1} the private key that is exclusively known to P. Given these keys and the corresponding algorithm, it is possible to *encrypt* a message m, denoted $\{m\}_{K_P} = m'$, so that only P can *decrypt* m' using his private key: $\{m'\}_{K_P^{-1}} = m$.

[4]The object might store the information in a form that is not easily accessible (e.g., encrypted or otherwise difficult to access). Since the object itself can access the information, all the necessary data has to be available in the object (e.g., the decryption key is coded in the object). A profound analysis (reverse engineering) of the object will hence, with high probability, allow to recover the information.

[5]Such objects have recently attracted a lot of attention in the context of Java applets (*Java* is a trademark of Sun Microsystems Ltd.) and ActiveX controls (*ActiveX* is a trademark of Microsoft Corp.). We will from now on simply refer to them as objects.

Some public key cryptosystems (notably RSA) also allow to use the same keys for *signing* messages. A message m signed by P is denoted $\{m\}_{K_P^{-1}} = m_{sig}$. The signature on m can be verified by everyone using P's public key: $\{m_{sig}\}_{K_P} = m$. If a message, which confers certain rights, bears the signature of a trusted principal it is also referred to as *certificate*.

3.2 Tamper Proofedness

The concept of tamper proofedness usually applies to a well-defined module, sometimes called black-box, that executes a given task. The outside environment cannot interfere with the task of this module other than through a restricted interface, which is under the control of the tamper proof module.

The actual construction of such a tamper proof module in the real world is extremely difficult, nevertheless, there are many applications that rely on them (e.g., payphones, debit cards, or SIM cards for GSM). Given sufficient time and resources, it becomes very probable that an attacker can violate the protection of the module [1]. Therefore, a tamper proof module should not protect information that is more valuable than the estimated cost[6] for breaking its protection. For the remainder of this presentation, we make the explicit assumption that the investment necessary to break our system is much greater than the possible gain.

4 The Overall Approach

In this Section, we will introduce the protocol that guarantees the integrity of the execution environment and protects the code and the data of an object executing in this environment.

We will first present the execution environment that we rely on and then describe the CryPO (cryptographically protected objects) protocol that uses it. Figure 1 gives an overview of the entities in the system and their interactions.

4.1 The Execution Environment

In order to achieve our goal, we need a special execution environment based on the notion of tamper proofedness, which we will call *tamper proof environment* (TPE). The TPE provides a full execution environment for objects, which can not be inspected or tampered with.

The TPE is a complete computer that consists of a CPU, RAM, ROM, and non-volatile storage (e.g. hard-disk or flash RAM). It runs a virtual machine (VM) that serves as execution environment for objects and an operating system that provides the external interface to the TPE and controls the VM (e.g., protection of objects from each other). Furthermore, the TPE contains a private key K_{TPE}^{-1} that is known to no entity other than the TPE – even the physical owner of the TPE has no information concerning this private key (see 4.2).

The TPE is connected to a host computer that is under the control of the TPE owner. This host computer can access the TPE exclusively through a well defined interface that allows, for instance, the following operations on the TPE:

- upload, migrate, or remove objects;
- facilitate interactions between host and object or between different objects on the TPE;
- verify certain properties of the TPE (such as which objects are currently executing).

Due to its implementation as tamper proof module and the restricted access via the operating system, it is impossible to directly access the information that is contained on the TPE.

[6] This cost can only be a rough estimation and is likely to decrease over time, when more efficient attacks become possible.

This property is ensured and guaranteed by the TPE manufacturer[7] *TM*, which also provides the object user *OU* with a certificate (signed by TM). The certificate contains information about the TPE, such as its manufacturer, its type, the guarantees provided, and its public key. The object provider *OP* has to trust the TM that the TPE actually does provide the protection that is claimed in the certificate.

4.2 Key Generation

One of the most important properties of the TPE that has to be guaranteed with utmost certainty is that the private key of the TPE is never disclosed. To ensure this property, the TPE could create the key itself (based on some random input). Using this approach, the key is never available outside of the TPE and, thus, protected by the operating system and the tamper proofedness of the TPE. This initialization can be done by the TPE manufacturer and subsequently the TPE will be prevented from ever disclosing this key or accepting a new one.

Other, more sophisticated approaches to create the pair of keys could be envisaged, which could also incorporate key recovery mechanisms (e.g., escrowed key shares).

4.3 CryPO Protocol

The CryPO (cryptographically protected objects) protocol transfers objects exclusively in encrypted form over the network to a TPE. Therefore, it is impossible for anyone who does not know the proper key to obtain the code or data of such an encrypted object.

The protocol is divided into two distinct phases. The first phase consists of an initialization, which has to be executed once before the actual execution of the protocol. The second phase is concerned with the usage of the TPE. The protocol is based on the interactions given in figure 1.

4.3.1 Initialization

In the initialization phase, the participants establish the trust relations that are associated with the different keys:

- the TM publishes its certification key K_{TM};
- the TM sends the certificate $Cert_{TPE} = \{K_{TPE}\}_{K_{TM}^{-1}}$ to the OU.

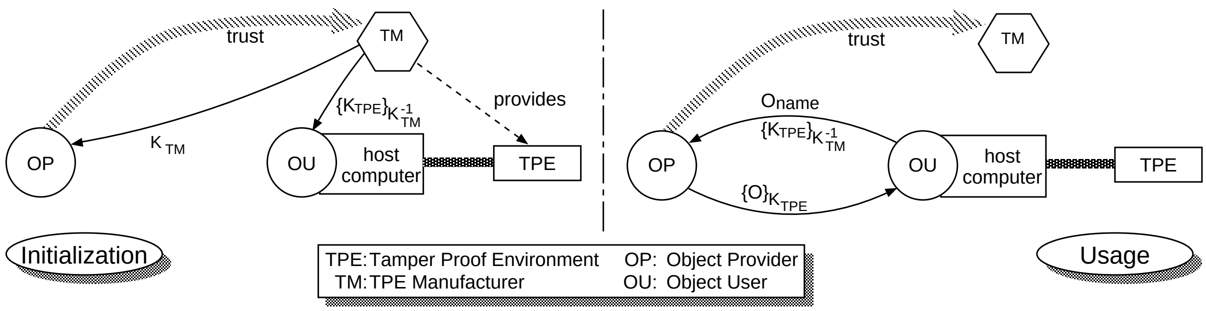

Figure 1: Overview of the CryPO protocol

[7] It would be possible to extend the model and introduce a certifier that takes some of the responsibilities (e.g. regular controls).

4.3.2 TPE Usage

After the participants have finished the initialization, they can execute the actual CryPO protocol:

- The OU sends a request to the OP that contains the name O_{name} of the object he wants to obtain together with the certificate $Cert_{TPE}$ of his TPE.
- The OP verifies the certificate $Cert_{TPE}$ to check the manufacturer and the type of the TPE, in order to decide if it satisfies the security requirements of the OP. Then the OP verifies other information associated with the interaction (such as OU's identity, payment, or others). If the OP is satisfied with these checks, it will continue with the protocol.
- The OP sends the object encrypted under the public key of the TPE, $\{O\}_{K_{TPE}}$ to the OU.
- The OU cannot decrypt $\{O\}_{K_{TPE}}$ nor can he do anything other than upload it to his TPE.
- The TPE decrypts $\{O\}_{K_{TPE}}$ using its private key K_{TPE}^{-1} and obtains the executable object O. Depending on the directives of the OU, O will eventually be started on the TPE and can interact with the local environment of the OU, other objects on the TPE, or even with remote applications.

4.4 Guarantees of the CryPO Protocol

The protocol described above guarantees the integrity of the execution environment. This guarantee is based on the trust relation between the OP and the TM. The OP trusts the TM to properly manufacture its TPEs so that the claimed guarantees hold, even without any physical control or access to the TPE. Thus, the objects are protected against manipulation and disclosure of code and data both in transit and during execution and are therefore safe from reverse engineering.

The OU is guaranteed that the object can only access the host computer through the specified interface. Any damage that could possibly be caused by an object is limited to the TPE and the restricted access it has to the user's host computer. By adding a digital signature on the object, the CryPO protocol could easily and transparently be extended to incorporate an object certification, such as those defined for Java or ActiveX [5, 7].

4.5 Additional Properties

Apart from the basic protection of the code and data of an object, the CryPO protocol can ensure other interesting properties that can be useful for certain applications.

4.5.1 Object Migration

The operating system of the TPE might allow an object to move to a different TPE. To achieve this, the TPE will obtain the certificate for another TPE, TPE′. The operating system will then check whether the other TPE satisfies the required properties. This is a complicated decision that depends on many non-technical issues, which are not within the scope of this presentation. The operating system will then stop the object (e.g., by calling a special method of the object that ensures that it is in a coherent state), marshal it into a transportable form (including all the code and the current data), and encrypt it with the public key $K_{TPE'}$ of TPE′. The object will then be sent to TPE′, where it will be loaded as usual and resume its execution.

The operating system of the original TPE can take further measures to ensure that the object will only exist as a single incarnation (if this condition is deemed necessary).

4.5.2 Message Confidentiality and Receiver Anonymity

The operating system of the TPE could allow objects on the TPE to receive messages. These messages can be encrypted with the public key of the TPE, so that no other entity can access the contents of such a message – not even the owner of the TPE. Thus, the confidentiality of the message content is under the exclusive control of the receiving object. It can make the content available to other entities or it can decide to not disclose it at all.

Furthermore, if the identity of the receiving object is also encrypted with the public key of the TPE, no entity outside the TPE can observe which of the objects currently residing on the TPE has actually received the message. Provided that the operating system of the TPE does not reveal any information about the result of the communication operation, then even if there is only a single object on the TPE, it might not be the receiver of the message. Consequently it is again the exclusive decision of the receiving object to identify itself as the receiver of the message. This approach can be used to implement receiver anonymity for the objects on the TPE. Possible attacks on the receiver anonymity by the owner of the TPE that consist of uploading an object, which pretends to be the legitimate receiver, but makes the message available to the user have not been studied yet.

Other interesting ideas are object expiration, where an object has a well defined time to life before it is eventually removed by the operating system of the TPE or the possibility to include a private key in an object so that it can sign the messages it sends.

4.6 Possible Problems

The proposed protocol has a weak point that we have to be very careful about. If the private key of a TPE becomes available to some user, that user can pretend to run a TPE (using the certificate for the TPE of which he has the private key), while he is actually capable to decipher everything that is sent to the TPE. This would violate all the security guarantees offered by the CryPO protocol with a slim chance of that user being discovered.

To alleviate this problem we could require regular inspections of the TPE hardware by some independent certifier. If we can ensure that an attacker cannot obtain the private key from the TPE without destroying it in the process, he would not be able to present the undamaged TPE to the certifier and thus, the duration during which damage could occur would be limited over time.

5 Application of CryPO to Mobile Communication Systems

Using the CryPO protocol introduced above, the solution to the problem of protecting the location information in mobile telecommunication systems as described in Section 2.1, is now straightforward.

We require the presence of a TPE in the underlying telecommunication system that satisfies all the properties introduced in Section 4.1 as well as a special object, which we will call *User Agent* (UA). A user, say Alice, can obtain such a UA from a trusted software vendor[8] and customize it to suit her special requirements. She will then encrypt the customized UA with the public key of the TPE and send it to her telecommunication provider. If Alice has previously established a contract with the telecommunication provider, the latter will upload the encrypted UA to its TPE, where the UA will subsequently start its execution.

The functionality of this UA is very similar to that of the HPC described in 2.1. Whenever Alice moves to a different cell, she will send a location update to her UA, encrypted with the public key of the TPE on which her UA resides (for optimization she can establish a shared session key with her UA and encrypt the data with a shared key algorithm). Hence, Alice's location is exclusively known to her UA, but not to the telecommunication provider. If another user, say Bob, tries to reach Alice, the telecommunication provider will query her UA in order to obtain her current location. The UA in turn can now apply the methods introduced in 2.1 to decide whether it should disclose the information.

Since we assume that Alice trusts the TPE to behave according to its specification (i.e., to not disclose any information about her UA) and that she also trusts her UA to not disclose her

[8] The telecommunication provider, which controls the TPE could require access to the source of the UA, in order to verify that the UA does not implement malicious behaviour (Trojan horse attacks).

current location without proper authorization, the goal of protecting the access to confidential information is again achieved.

5.1 Advantages of the Approach

Compared to the approach proposed in [9], the one introduced above has several advantages:

- the TPE is under the administration of the telecommunication provider, which will ensure its reliability. The telecommunication provider can also verify the code of a UA to ensure its reliable operation.
- the TPE is located in the core of the telecommunications system and can be interconnected with high speed links.
- the UA can be migrated to a different TPE (see 4.5) if the user moves to a different network for an extended period of time.
- the TPE can be physically protected at the site of the telecommunication provider, which can probably also put better software protection (e.g., firewalls) in place.
- the TPE will host many UAs (on the order of several thousand or more). If the TPE allows receiver anonymity (see 4.5), even the telecommunication provider would have to do heavy traffic flow analysis to identify the called user. A more detailed analysis of this issue is necessary to allow a quantification of the incurred protection. If the underlying telecommunication system provides MIXES [3], these can also be used to further protect the location information.
- a recent approach to define the next generation of telecommunication systems is TINA [8] (Telecommunications Information Networking Architecture), which is developed by a consortium of many important telecommunication providers and equipment manufacturers. In TINA there exists the notion of a User Agent [13], which is an object that has very similar tasks to those described above. However, the UA in TINA executes under the control of the telecommunications access provider (in TINA this entity is called *retailer*). We consider it possible to integrate our approach into TINA and will pursue this issue in our future work.

5.2 Feasibility

We assume that the TPE is a quite powerful computer with a special operating system. The TPE is physically protected with a special hardware that can effectively be sealed to detect tampering, is under continuous video surveillance, and subject to challenge inspections by an independent organization (such as the TÜV[9]). Since a telecommunication provider does only need a small number of these installations, we consider the cost for its realization quite realistic.

As explained in [1], such an installation is conceivable and can even resist massive attacks. Again, a thorough analysis of the remaining risks has to be undertaken, but this is not within the scope of this presentation.

Another problem exists with legislation that requires access to the information we try to protect. To resolve this dilemma, UAs could be equipped with a trapdoor that allows to access its internal data. Since the telecommunication provider has to approve of UA implementations, it could easily refuse those implementations that do not provide this trapdoor. Obviously, this functionality would have to be cryptographically protected with a highly secure key that would be used to sign such orders issued by a judge.

[9]The *Technischer Überwachungs-Verein* is an independent, government-approved expert appraisal and inspection organization.

6 Conclusion

An approach to protect confidential information in mobile communication systems has been presented. The approach takes advantage of the fact that oftentimes, information is not actually needed and restricts the disclosure of the information to the required minimum. The underlying idea, which allows to retain some control over information handed to a different entity, is sufficiently general to be applicable to other problems with similar constraints.

The approach is illustrated with the example of protecting location information in a mobile communication system, which uses simplified concepts from GSM (Global System for Mobile Communications). It is compared to another approach from the literature [9].

We further define the problem of protecting mobile objects against manipulation and disclosure and present a solution, which consists of a tamper proof execution environment and a protocol that provides a comprehensive protection for the objects executed on this environment. This can be used to protect objects against reverse engineering, which is an interesting application in the context of Java and ActiveX.

Acknowledgments

I would like to thank Sebastian Staamann for the discussion that spawned the presented idea.

References

[1] R. Anderson and M. Kuhn. Tamper resistance — a cautionary note. In *The Second USENIX Workshop on Electronic Commerce Proceedings*, pages 1–11, Oakland, California, November 1996.

[2] G. Brassard. *Modern Cryptology – A Tutorial*, volume 325 of *Lecture Notes in Computer Science*. Springer Verlag, 1988.

[3] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, February 1981.

[4] H. Damker, H. Federrath, M. Reichenbach, and A. Bertsch. Persönliches Erreichbarkeitsmanagement. In G. Müller and A. Pfitzmann, editors, *Sicherheit in der Kommunikationstechnik*. Addison-Wesley, 1997.

[5] J. S. Fritzinger and M. Mueller. Java security. White paper, Sun Microsystems, Inc., 1996.

[6] A. Herzberg and S. S. Pinter. Public protection of software. In *Advances in Cryptology: CRYPTO'85*, pages 158–179, Santa Barbara, California, August 1985.

[7] P. Johns. Signing and marking ActiveX controls. *Developer Network News*, November 1996.

[8] G. Nilsson, F. Dupuy, and M. Chapman. An overview of the telecommunicatinos information networking architecture. In *TINA'95*, pages 1–11, Melbourne, Australia, February 1995.

[9] A. Pfitzmann. Technischer Datenschutz in öffentlichen Funknetzen. *Datenschutz und Datensicherung (DuD)*, 17(8):451–463, 1993.

[10] RSA Data Security, Inc. *PKCS #1: RSA Encryption Standard*. RSA Data Security, Inc., November 1993.

[11] B. Schneier. *Applied cryptography*. Wiley, New York, 1994.

[12] New York Times. U.S. workers stole data on 11,000, agency says, April 6, 1996.

[13] TINA Consortium. *Service Architecture, Version 4.0*, October 1996. http://www.tinac.com/96/sa96_public.ps.

[14] I. S. Winkler. The non-technical threat to computing systems. *Computing Systems, USENIX Association*, 9(1):3–14, Winter 1996.

[15] P. R. Zimmermann. *PGPfone Owner's Manual*. Pretty Good Privacy, Inc., July 1996. http://www.pgp.com/products/fone-docs/fone_01.cgi.