The MIT Smart Card:

A Pseudo-Anonymous, Token-Based Authentication System

Recitation: Rivest TR 1

Nonce: 050647

Mark Ethier
msethier@mit.edu

Michael Freedman
mfreed@mit.edu

Ajay Kulkarni
kulkarni@mit.edu

April 27, 2000

Abstract

This paper presents a smart card system design for use within the MIT community. The proposed design employs a token-based access-control model, where access is granted by group membership and not by individual identity. Several encryption and authentication methods ensure security for verification and token/key distribution. Users perform access-control administration in a distributed and delegated manner. The design addresses tradeoffs between functionality and privacy while attempting to ensure a secure and easily maintainable system.

Contents

1 Introduction

The Massachusetts Institute of Technology has decided to explore the use of a secure smart card to both replace its current magnetic strip card system and to provide additional conveniences to the MIT community. Currently to control access to resources, MIT employs an array of security methods ranging from combination and metal key locks to magnetic cards. This current system is susceptible to many security and impersonation attacks and does not allow easily enforceable administrative policies. A widely-used smart card system would remedy many of these problems.

Smart cards resemble magnetic strip cards but also include a microprocessor, memory, and a method of external communication. A smart card also has many advantages over a magnetic card: the ability to store information, to more securely transmit information, and to perform operations on information.

With the added benefits of the smart card come serious concerns about the protection of privacy. As the smart card system could potentially maintain more detailed records of card use, the fear of "big brother" watching each individual's actions becomes apparent. This paper proposes a smart card system that, in all but some rare cases, maintains a satisfactory level of anonymity by assigning tokens to groups and not to individuals.

Smart cards allows increased security. The ability to store information and perform calculations, both within the card and card reader, enables three main secure protocols: verifying cards by card readers, updating permissions on smart cards, and updating tokens and certificate authority public keys on card readers.

With the added functionality comes a more complex administration system. For simplicity, users perform the administration of the system in a distributed and delegated manner. Sub-authorities are granted permission over their own cards and certificate authorities. Furthermore, policies are flexibly determined by those who are closest to the administration of the actual resources.

The remainder of this paper is organized as follows: section 2 details the design requirements and various options for a smart card infrastructure; section 3 presents a detailed description and discussion of the smart card system design; section 4 highlights weaknesses and describes extensions to the current design; and section 5 concludes the proposal.

2 Design Criteria and Considerations

The smart card system must at a minimum provide the following:

- Individual Authentication
- Group Management
- Authentication of Group Membership
- Delegation of Group Membership for periods of time

The system should be able to handle methods of providing limited access to dormitories, labs, or other buildings, authenticating users to various campus services, and demonstrating proof of memberships in various subgroups.

A smart card system has several advantages over earlier systems when performing these operations. Previously used metal key and magnetic strip card systems merely serve as a storage of information: the notches on a key represent its immutable information; the magnetic strip contains the card's mutable information. On the other hand, a smart card is a microprocessor in addition to a mutable memory, therefore able to hold state and perform computations on the card.

The ability to perform operations enables the smart card to perform a larger number of tasks: to receive and transfer information; to perform operations on that data; etc. Potential data operations include encryption or decryption of the data, thereby allowing for a greater security of transactions. The smart card also includes intricate storage memory, allowing the card to store much more data than the earlier systems. By relying on these internal devices for functioning, and not a physical characteristic or external property, smart cards are also more resilient to physical damage than earlier card systems.

2.1 Privacy

Gradually, more and more tasks of an individual's routine are being enhanced by current technology. What started as an electronic mail system is now being used for news, shopping, even banking. These enhancements bring with it a greater ability to track an individual's routine, which in turn brings up concerns with protecting that individual's privacy. The primary reason why smart cards introduce privacy concerns can be found in their name: smart cards are "smarter" than their predecessors. But how smart should smart cards be?

The primary smart card predecessors, namely the metal key and magnetic strip systems, by default only protect a specific level of privacy. With metal keys, privacy is essentially dependent on the number of individuals possessing the key. Different metal keys may fit the same lock, but it is impossible to detect which key was used. With a magnetic strip card, no privacy can be enforced through the technology, as each card can be traced to a specific user. While privacy may be protected within the policy of the administrators of the magnetic card system, that policy may change. Moreover, anyone who breaks the system could also compromise individuals' privacy.

Analyzing potential smart card uses reveals how a smart card system can allow for the protection of a varying degree of privacy within its architecture. Individuals primarily use smart cards in accessing rooms, facilities, and devices. The system could maintain a record of each use by each user; similarly it could not record any information about the smart card use. The different methods of recording can be categorized into four distinct levels: recording the full identity of an individual and use; recording a group name and use; recording a use; not recording any information.

The smart card system could be designed to permit recording at all four levels, or to restrict recording to a certain level. That is, should the system trust individuals – both current administrators and future potential administrators – to make correct policy decisions? Or should the system restrict how individuals may administer the system, based on what is decided to be a "correct" policy today?

Even if the smart card system administrators, in this case members of MIT, were trusted, a certain degree of recording may be prohibited to minimize the personal and physical cost of MIT's records' security being compromised.

2.2 Security

Lessons learned from previous security technologies suggest design goals for the security of a smart card system. Keys have existed for centuries in various forms; currently, lock technology has advanced to make picking and unauthorized access more difficult. However, the mere physical possession of a key is sufficient to provide access and thus, by extension, membership within a group. Therefore, lost, stolen, or copied keys provide a large security risk.

Magnetic strip card systems provide similar security risks as their metal key predecessors. Physical possession of a card is sometimes sufficient to attain its benefits and permissions, although PINs are used in some magnetic cards. Furthermore, cards can be counterfeited rather easily by writing or copying the magnetic domains.

The limitations of older key and magnetic strip card technology suggest central security requirements for smart cards:

- Physical possession is not sufficient to gain the card's benefits.
- The method by which identity or proof of membership is demonstrated is not easily counterfeited.

User authentication should be drawn from one or more of the following characteristics: something the user *has* (physical possession of the smart card), something the user *knows* (PIN number, password, or biographic information), and some *physical characteristic* of the user (fingerprint, retina scan, or other biometric information.) Our design should incorporate both the physical possession of the smart card as well as knowledge of some PIN number during card use. Lacking the means of biometric authentication is an acceptable tradeoff: while it certainly adds another layer of security, the current cost and size of such technology is prohibitive. Future versions of the card could easily include biometric authentication hardware, without changing the security protocols: another step in user authentication would merely be added.

Smart card authentication is necessary to ensure that proof of membership is only accepted when presented by an authorized source. System security is necessary for two types of card usage: first, interactions between smart cards and card readers that are used to provide physical access or demonstrate permission for services; second, interactions between smart cards and host systems that authenticate the cards with certain permissions.

All communication protocols between the card, card reader, and host authorities should be resistant to both passive and active adversarial attack. Namely, an adversary should not be able to gain unauthorized access from a card reader by presenting falsified information. Nor should he be able to gain sufficient information to impersonate an authorized user. Similarly, an adversary should not be able to convince a host authority to issue it undeserved permissions. Cryptographically secure primitives and physical security measures should be used to fulfill these security requirements.

2.3 Operation and Management

Currently, most facilities are accessed by means of combination locks and keys. The administrator of a group (whoever holds the keys) chooses the group's policies by deciding who should have a copy of the key. In the case of a combination lock, each person who knows the combination can choose who has access. The current system has the advantage that each group can choose its own level of security. A new system that uses smart cards should not impede this freedom but should impose a more formal definition of policies once specified. The design should allow administrators to flexibly assign privileges accordingly as best suits their group.

Operation and management follows a hierarchical system of databases and certificate authorities. The central MIT certificate authority can access several different databases that correspond to specific groups, which administer their own specific security policies and assign permissions. Likewise, a group at MIT should be able to take full control over the administration of the certificate authority and the associated policy database. With these elements under their control, a group also has control over the card readers and token distribution. This distinction allows a modular security model and limits the extent of an attack on any subsection.

Resources – such as access to a door or machine – should be organized in a hierarchical manner so that an administrator may choose to delegate permission of sets or single objects. For example, an administrator of a building needs to delegate responsibility to the administrators of individual labs. Without a hierarchical system, the building administrator (BA) would have to allow a lab administrator (LA) explicit access to every individual door and piece of equipment in some lab A. If that LA was replaced by another user, the BA would have to revoke each permission from the old LA and grant each permission to the new LA. With a hierarchical system, the BA could simply organize all resources in the lab under Lab A. The BA would assign this permission to the LA and revoke the permission of the former LA.

Unfortunately, Moira on Athena is not adequate for managing groups. Moira is designed as a distributed system administration tool and is not considered secure. Athena can be used for obtaining and authenticating users (authentication occurs through Kerberos [6]), but a secure group, policy, and resource management system will have to be implemented by another information system.

Delegation can take two forms: an administrator delegates administrative permission over resources to other users; and a user can grant fellow users access to resources. The first is dictated by administrative permissions where the second is dictated by policies. Administrative delegation allows a hierarchical system to reduce complexity. Access delegation with smart cards parallels the act of copying metal keys for other users. This type of delegation should not be achieved through a token "borrowing" scheme: users should not be allowed to lend their tokens or certificates to other users. In the proposed system, a new user should be granted permissions by the administrator of a resource.

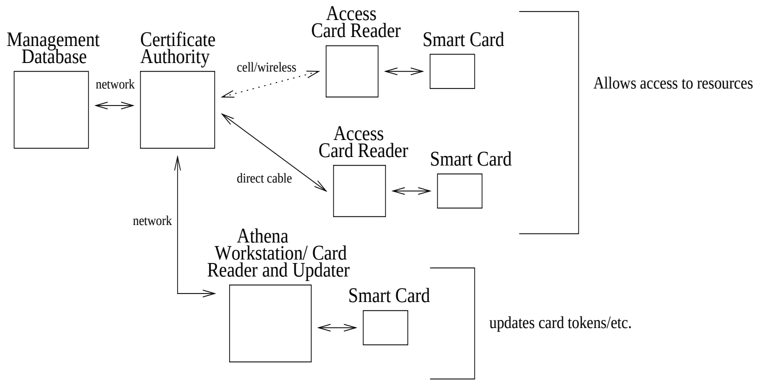

Figure 1: Communication Architecture

3 Design

This section details the design and implementation of the smart card system.

3.1 Architecture

The proposed card system is divided into four main sections: the smart cards; the card readers; the group, resource, and policy management (GRPM) databases; and the certificate authorities (CA). A smart card will be comprised of the following: a microprocessor, memory, a numeric keypad, a battery, and LED indicator lights. The card readers will be powered by either DC or AC power and will contain a network-interface card or cell modem, a microprocessor, memory, internal clock, and LED indicator lights. Several different types of memory are used in these systems: RAM for in-memory calculations, PROM for storing tokens and certificates (and CA keys on readers), and ROM for securely storing public-private key pairs. The GRPM databases will be constructed with a database and an interface protocol. The CA is a trusted third party certificate authority.

Card readers and the CA will transmit information through a communication network. If necessary, new cables could be laid down connecting all the card readers, but most card reader locations would probably have a nearby network connection.

Short-wave radio was initially considered as a communications medium. Such a system would minimize the need for establishing a network infrastructure. Moreover, communication between CA and each reader is infrequent and very low-bandwidth. Short-wave radio would easily enable broadcast transmissions, but point-to-point communication would require specifying a naming scheme or using different radio frequencies. This method is also subject to jamming, interference attacks, and snooping.

A hard-wired network makes such attacks more difficult, as an attacker would require access to the network to interfere with or listen to signals. A network allows for both point-to-point communication (used for token updates) and for CA broadcasts to all card readers (used for time synchronization). Where a network connection is not available, or is not readily installable, a cellular modem or a wireless ethernet connection may be used. The cellular modem has a larger range of use and requires less infrastructure, but it prohibits the card reader from being able to listen on a time synchronization broadcast. The CA would have to specifically update each cell modem card reader individually. Again, as card reader/CA communication is infrequent, it would not require much bandwidth, thus enabling the use of an existing network and keeping the cost of the use of a cell modem low. Certificate authorities, therefore, need to associate each card reader with some means of communicating with the card reader, be it network IP address, dedicated network name, or cellular modem number.

The GRPM database will store: groups of users, groups of administrators, groups of resources, policies, and groups (users, administrators, resources) associated with each policy. As the database's security is critical to the overall security of the system, implementors must take adequate precaution in preventing any data compromise.

Security policies will include: resource lists, valid-from and expiration dates, permitted time of access, logging level, delegation policy, renewal policy, and a token. Tokens uniquely identify security policies.

3.2 Privacy

The smart card system should preserve a certain degree of anonymity: specific individual logging should not be permitted. Although such a restriction could be implemented within policy, it is better to establish the restriction within the design itself – an inherent restriction disallows any person from being able to easily monitor individuals.

When a card accesses a card reader, it receives a form of the list of accepted tokens from the card reader. The card then sends a message including its public key and a form of one of the accepted token(s). A card is verified by a card reader by verifying if it has a correct token. For card readers that need to restrict access for a card for a specific number of times – e.g., a parking lot card reader – the card reader also checks the card's public key to ensure that that card is not currently in use of the facility that implements the card reader.

Token verification can occur in two different ways, dependent on which type of token is implemented. In a list based token model, the card reader would contain a list of all acceptable tokens. Each acceptable token would correspond to a specific individual. When a card presented its token to the card reader, the reader would look up the specific card token in its list for authentication.

The main problem with this model is that the card reader is able to recognize which token the card contains, i.e., to which user the card belongs. This information could be used to track users. Also, the size of this list could become very large, depending on the type of access the card reader grants. For a large dormitory, for example, the card reader would need to maintain hundreds of individual tokens. Obviously, the card reader could merely be a front-end that communicates access requests to some centralized database. Still, logging and snooping could be easily used to determine an individual's access times and patterns.

Being able to maintain group specific tokens is be a better way to use tokens. This way, each card would contain tokens specific to the groups to which the card owner belongs. Each card reader would then similarly contain tokens corresponding to a list of authorized groups. For most cases, only a few tokens would be maintained by a card reader, as usually only a very few groups would have access to a specific small area.

Note that although the card reader receives the card public key during transmission, this knowledge does not necessarily compromise the privacy of that specific card. The CA signs public keys, and distributes them to individual cards. However, if a card reader cannot look up a public key and receive information to which card that key belongs – that is, if the CA itself does not maintain a mapping of public keys to users – then the card reader can not easily trace the public key to a card.

However, the card reader can still keep track of which public keys have been presented; indeed, some card readers need to maintain this information. Likewise, if a card reader is bogus, an adversary can physically monitor a card reader to map public keys to user identities. Therefore, the card's access is not fully anonymous at the individual card level. The individual's identity is as anonymous as it is difficult to trace public keys to specific cards (and thereafter, to specific individuals.)

This is the model implemented in this proposal. Monitoring the card reader's information then provides information on which group and which public key accessed the card reader at which time, not specifically the user identities. For the most part, this logging provides a satisfactory level of anonymity. In addition, logging can be used in case accesses to a card reader need to be reviewed: e.g., for investigating trespassing, theft, or other forms of crime. The log could provide the information whether someone was in the room during the crime, which group this person belonged to, etc. While a log of which individuals used the card reader could perhaps aid an investigation more than the proposed type of log, the proposed design provides a tradeoff that does not compromise an individual's privacy.

A better smart card system would prohibit recording the card's public key information – a card reader would only be able to verify a submitted token and public key. Although this may not preserve complete anonymity (a defined subset of users who could have used the card reader may exist), it protects a greater amount of privacy than the current proposal. Although this method would provide a less detailed audit trail, the privacy advantages outweigh this loss. However, to enforce this level of anonymity, the design would require more elaborate methods of encryption (see section 4.2) than were permitted by the specification of the smart card system.

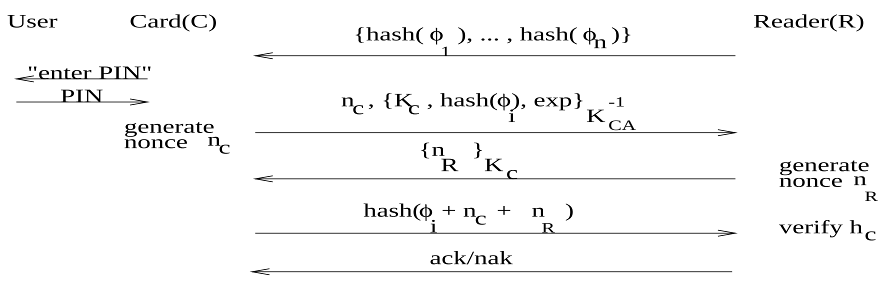

Figure 2: Smart card authentication to a card reader

3.3 Security

This section specifies the design of security protocols between smart card system components. The basic use of the smart card can be described in several steps.

3.3.1 Demonstrating Proof of Membership: Cards and Card Readers

The system design has focused on decentralized authentication, while providing pseudo-anonymity on the token-based access system. Therefore, using tokens for access-based control allows verification on the card reader level, as opposed to queries being actively sent to some centralized database system. For our design, the card readers do not log or transmit any uniquely identifying documentation, except for special applications as described in section 3.3.4.

The smart card (C) will make use of a digital certificate signed by the certificate authority (CA) and a token ϕ_i to demonstrate access permission to a card reader (R). The asymmetric key notation used is the following: K_P is the public key of some principal P, K_P^{-1} is the principal's private key.

The smart card authentication protocol is shown in Figure 2.

1. When the smart card approaches some card reader, it is initially presented with a list of hashed tokens that may be used to gain access.

2. The card challenges its user to provide a PIN for user authentication. If the PIN is not correct, the procedure halts.

3. Upon verifying the user, the card transmits a provably secure [1] pseudo-random $nonce_C$ and its digital certificate signed by the CA, which includes the public key of the card, a hash of the chosen token ϕ_i, and an expiration date *exp*.

4. The card reader verifies that a trusted CA signed the certificate, that the $hash(\phi_i)$ corresponds to an accepted token for some given parameters (e.g., day of week, time), and that the expiration date has not yet passed.

5. Upon verification, the card reader responds with its own random $nonce_R$, encrypted by the card's public key K_C.

6. The card then decrypts the reader's nonce and takes the the hash h_C of $(\phi_{i_C} + nonce_C + nonce_R)$, sending this to the reader.

7. The reader verifies that $h_C ==$ hash$(\phi_{i_R} + nonce_C + nonce_R)$, acknowledges the card, and grants access accordingly.

An advantage of this design protocol is simplicity. A communication link is not required between the reader an some central source, thereby reducing verification time. The hash values of tokens in the initial (pre-nonce) stages can be pre-computed a single time for each token. The only bit of encryption takes place after the card transmits its public key, thereby greatly minimizing any necessary public key infrastructure.

Security relies on the dual nature of verification, using both a certificate and token. Because only the hash of tokens are sent between card and reader in initial stages, a listener cannot overhear plaintext tokens. The card does not attempt to match acceptable tokens to its list of tokens until after user verification, stopping users from easily determining the access granted by a card if they do not have a proper PIN. The protocol ensures that the card is the proper certificate holder by authenticating the card's key by the CA, and using public key encryption to transmit the reader's nonce. Furthermore, during later stages of the verification process, the hash uses nonces specified by the card and reader during the challenge stage. Therefore, the user is protected against replay attacks by a listener. Even if the bogus card reader records h_C, an adversary cannot easily use h_C for an impersonation attack.

Consider an on-line impersonation attack that utilizes bogus card readers and smart card adversaries that can communicate. Some smart card user Alice attempts to use a bogus card reader Lucifer. At the same time, an adversary Eve attempts to use a valid card reader Bob. As Eve needs to commit to some nonce before she receives Bob's response nonce, she cannot collude with Lucifer to use h_{Alice}, as the smart card nonces would be different.

The security of the design, therefore, rests upon the security of the message digests and digital certificates. For taking message digests, we shall use the SHA-1 hash function, specified in [9]. The function produces a 160-bit message digest that is collision-free (one collision every 2^{80} brute-force attacks) and secure against inversion attacks. We shall use the RSA cryptosystem for digital signatures on the (key K_C, token ϕ_i, exp date) certificate. Provided the private key of the CA is not revealed, this aspect of system security relies on the factoring problem inherent to RSA.

3.3.2 Gaining Individual Permissions: Cards and Certificate Authorities

The smart card system requires a secure method by which the card can get ϕ_{i_C} and $cert_{i_C}$, thus gaining proof of membership to some group represented by the token ϕ_i. This section describes a communications protocol between smart cards and certificate authorities, based on SSL [5].

1. The user physically presents his smart-card to an untrusted workstation or special-purpose, networked card reader.

2. The smart card requests the user's PIN number, to be typed into the smart card keypad directly.

3. Upon PIN verification, the smart card accesses some trusted certificate authority (through the workstation/card reader), and instantiates an SSL connection.

4. The CA responds with the proper "Server Hello" and "Server Certificate" messages. The smart card continues with the typical SSL handshake protocol, yielding a `master_secret` to generate symmetric keys between card and CA. All further communications are encrypted with these keys.

5. The smart card transmits its encrypted personal certificate (which includes its public key $\{K_C\}_{K_{CA}^{-1}}$), and a request for updated tokens and certificates.

6. The CA sends the card an update of all tokens ϕ_i and signed certificates $\{K_C, hash(\phi_i), exp\}_{K_{CA}^{-1}}$.

This protocol allows the system to rely on the security of SSL for communications across untrusted lines – through the workstation, across network connections – as the SSL connection is established directly between the card and CA. The system presupposes card knowledge of the CA's public key, as well as a personal public key adequately authenticating itself to the CA. Given some initial K_{CA} and $\{K_C\}_{K_{CA}^{-1}}$ when the card is issued, these can be updated using the same protocol. Since current popular Internet browsers do support smart cards as add-ons [10], this implementation does not require completely new software for SSL connections on the workstations.

The protocol is also specified to consider untrusted workstations. Indeed, a workstation is not directly required for updating tokens/certificates. As the considered smart card does not have any significant display capabilities and adequate user interface, the CA always sends an update of all necessary documentation (i.e., all new tokens or nearly-expired certificates.) The screen display capabilities of the workstation do allow the smart card to inform the user of the current status of its certificates, if so desired.

The adoption of trusted workstations or more complex smart cards would allow users to selectedly request and update tokens-certificate pairs. This design yields an advantage in user privacy. Using selection, the

CA can log which tokens the user actively requests. With this design, the CA only knows of which tokens the user receives. Therefore, the user can repudiate ever seeking (or even knowing of) specific tokens.

3.3.3 Ensuring Physical Security: Tamper Resistance

Both the physical smart cards and card readers must be resistant to physical break-in. In other words, an adversary should not be able to "open" the system components and read tokens from PROM memory or private keys from ROM. First, tamper resistant hardware should be used to stop attackers from gaining such knowledge. Such hardware and associated secure OS is an ongoing research area, an example is the Hitachi H8/3112 MULTOS chip [8] as used by Mondex cards [7]. Second, if tampering is suspected, the reader should attempt to inform the proper authority over its network connection, so that tokens and keys can be properly revoked and replaced.

3.3.4 Limiting Use: Special Access Facilities

A limited use facility usually includes once-at-a-time and pay-as-you-go access. The described card system does not support pay-as-you-go services, but does support once-at-a-time uses. For example, a parking lot wishes to prevent "pass backs" of valid cards: a user opens the gate to allow their car into the lot, then passes the card back to another user to allow her car access as well. To prevent such misuse, special once-at-a-time card readers should store state from card accesses, as the card itself cannot be trusted to store such information.

1. The smart card and card reader perform the standard authentication procedure (section 3.3.1).
2. During the authentication, the reader stores a hash of the card's public key K_C.
3. When the card is used to exit the facilities, $hash(K_C)$ is removed from reader memory.
4. If the card attempts re-entry (i.e., the presented certificate authenticates a public key matching one stored in memory), access is denied.

There are several ways to attack such a security system. First, users can attempt to simulate exits: using a card but not actually removing a car. Weight sensors should be added around the gates to prevent this attack. Second, an attacker can attempt to reveal the identity of individuals by examining logged information. The hash of the public key is stored instead of the plaintext so that logged information cannot as easily be mapped to individual identity. Obviously, this information is available during the actual verification procedure, an attacker can merely exhaustively match the hashes of permitted individuals to determine identity, or an attacker can simply see which cars or users are present in the limited-access facility. Still, our system attempts to restrict misuse and minimize the disclosure of public keys.

3.4 Key and Token Distribution

Three primary components are involved in the management and distribution of keys and tokens: the certificate authority, the card reader, and the smart card. Each of these components use asymmetric keys for communications.

The private key of the smart cards and card readers should remain secret and immutable over the life of the component. One problem is physical access to card or reader memory through tampering. This design should make use of tamper-resistant hardware, as briefly discussed in 3.3.3. Smart card and card reader manufacturers should also not be trusted to assign and preserve private key information. The host authority itself (i.e, MIT) should initially assign a private/public key pair, to be written to ROM on the cards and readers. Changing a key involves replacing the ROM with memory containing a new key pair.

Card reader management makes use of an untrusted communications link between card reader and trusted certificate authority server. The distribution of tokens is required to refresh old tokens on card readers, replace tokens in the case of compromise, and issue new tokens when new groups are added to card readers. The card reader clock is also updated by the CA, to ensure time synchronousness for determining card certificate expiration. The clock does not need be accurate more than a few minutes.

In our model, certificate authorities have *jurisdiction* over card readers. In formal BAN logic [2] notation, $CA \Rightarrow R$. Time updates are periodically sent out by the CA or requested by readers: these updates should not need to occur more than once every day or so. Tokens are distributed or updated by the CA sending out an encrypted and signed message with the old tokens ϕ_{old} to be replaced and the new tokens ϕ_{new} (i.e., R sees message M – $R \triangleleft M$). Including the old token in the message M protects against replay attack by ensuring freshness: $\sharp(M)$. The card reader responds with an ACK/NAK message whether the message was valid: the signature correct, message able to be decrypted, and old tokens still valid. The reader can pre-compute the hashes of its tokens at this stage, to be used for the card verification process. Formally expressed, token distribution occurs as follows: given $M = \{\{\phi_{1_{old}}, \phi_{1_{new}}\} \ldots \{\phi_{n_{old}}, \phi_{n_{new}}\}\}$,

$$R \triangleleft \{M_{K_R}\}_{K_{CA}^{-1}}, \sharp(M) \longrightarrow R \text{ believes } M$$

Similarly, this link can be used for the CA to introduce other trusted CA's. Given $M = \{K_{CA_2}, \text{timestamp}\}$:

$$CA_1 \Rightarrow R, R \triangleleft \{M\}_{K_{CA_1}^{-1}}, \sharp(M) \longrightarrow CA_2 \Rightarrow R$$

3.5 Group Management

The section describes the management of groups within the system. We have previously described the method by which smart cards are validated on card readers via tokens and certificates, as well as the process of updating tokens on cards and readers. Easy configuration and management of these groups and resources is another crucial aspect of the system's design.

3.5.1 Permissions Hierarchy

There are two primary levels of permission. The highest level is defined by the certificate authority and policy database. Groups can maintain their own CA and policy database. All cards carry signed public keys from the MIT CA to prove identity, but card readers and cards can have tokens from any CA. A group that has control over a CA has control over the distribution of tokens and thus the policy of the entire sub-system. Users wishing access to these sub-system resources receive tokens from these individual certificate authorities, after verifying these CAs are authenticated by the central MIT CA.

The second level of permissions is defined at the CA and policy database level. Within the policy database, some class of super-users exist that have access to modify all users, resources, and policies encompassed in the group's domain. They bestow the first permissions to determine policy for an appropriate object hierarchy to the administrators of an upper echelon of an organizational scheme. (The scheme could possibly be organized by buildings, departments, etc.) The second tier administrators then delegate permission to subgroups below them and so on. Administrators can grant administrative permissions to others.

3.5.2 Policies

Each sub-level of resources has one or more policies associated with it. Policies are not inherited and only administrators are allowed to create and edit policies. Users are not assigned resources directly, instead administrators are associated with resources, and subgroups of users or individual users are allowed access to resources through policies created by administrators. If a user has multiple policies, he has access to the union of the permissions.

3.5.3 Administration

The design includes the following types of system administration:

- **Configuration:** The interface of the group management system should be display independent. That is, it should define a standard XML-RPC type protocol [12] for interfacing with the database. This should include standard Kerberos [6] authentication by Athena, but should not dictate formatting of the GUI. The protocol interface allows an independent implementation of the front-end display.

- **Delegation:** Administrative delegation is determined by the administrator of a resource as described in section 3.5.1. Access delegation is determined by a policy creator. In the metal key or combination lock system, a user can elect to have someone borrow his key or to tell someone the combination. That facility still exists in the smart card system, but it is performed in a more formal and secure way. If the administrator decides that access delegation is allowed in a policy, any user under that policy can delegate their level of access to another user. Using the protocol interface, the new user obtains a copy of the delegated policy. The new user is associated with its delegator in the GRPM database.

- **Revocation:** Revocation can be difficult in the described authentication system. When an administrator attempts to revoke a user's permissions, he will be prompted with statistics about the number of people that will be affected. At this point, an administrator can either choose to revoke permissions actively or passively. An *active* revocation is almost instantaneous. Although abstractly the administrator is only revoking permissions for one person, the system actually must revoke a security policy, create a new policy, and re-grant permission to all users except the removed user. In practice, this corresponds to new tokens being issued to each card reader, and each user obtaining a new token-certificate pair. With *passive* revocation, a user's certificate is not renewed when it expires. Since each certificate has an expiration date, the user will automatically lose access after some period of time.

- **Compromise:** The main protection against a smart card being compromise is through a PIN, as explained in section 3.3.1. Dealing with a compromise is a similar problem to dealing with revocation: the course of action must be decided on a case by case basis. For example, consider a user that lives in a dorm yet also works at a highly sensitive research lab. If the user reports his card stolen or missing and is uncertain whether his PIN was compromised, the administrators of each group would be notified. Each would make a decision based on that group's interest. The dorm administrator would probably just allow a passive revocation of permission, but the lab administrator would probably perform an active revocation. In the case of a card that is taken, used, returned without the card holder's knowledge *and* the PIN is compromised, there is no means in the system to deal with this attack. However, this weakness is not something that was introduced with smart cards: the same sort of borrow-attack would also occur with magnetic cards or metal keys, possibly even without the benefit of a PIN.

4 Weaknesses and Extensions

This paper describes a token-based smart card system design with fairly robust security. This section recognizes a few weaknesses in the design, as well as suggesting extensions for added security and/or privacy.

4.1 Delegating Access without Permission

The proposed smart card design stores public-private key pairs in ROM, generated and signed by the certificate authority. If an adversary determines a method of writing private keys, he can collude with another insider to attack the access-restriction of some resource. Namely, users have certificates that refer to their specific public keys. If a user can copy one of their card's token, corresponding certificate, and private key to a second card, that card gained permission to some resource. This problem is not due only to the token-based structure of our system. Even in a list-based model, a card with another authenticated public-private key pair would be able to falsely gain access. This delegation attack should not be permitted.

The system's security to such an attack is based on the strength of tamper-resistant hardware. Private keys are stored in ROM so that the keys cannot be changed without replacing the memory in real smart cards. The danger arises from bogus card that permit such an operation. In practice, card readers should be able to recognize valid cards by some system (not individual) unique hardware, although these techniques approach "security through obscurity." The hardware and card OS should be secure such that private keys are not wrongly divulged nor persistently stored in accessible RAM. Smart card companies such as DataKey [10] use such tamper-resistant hardware for security.

4.2 Increasing Anonymity

This design does not currently support the level of anonymity desired. The card reader uses the card's public key to verify that the card is the certificate's proper holder. The specified design has verification occur solely on the card reader without any logging taking place, notwithstanding hashed key logging for limited use facilities.

The design attempts to enforce privacy by specifying that the certificate authority should not log public key information of cards. Nor does the protocol allowing outside sources – other users, card readers, and so on – to query the certificate authority for a public key's user identity, if such was improperly logged. Still, as long as the ability to create this mapping exists and the public key is transmitted to the card reader, the system's privacy is dependent upon the CA's security as a trusted third party. Furthermore, merely monitoring the physical use of a card reader can yield similar key-identity information.

An improved design would make use of a blind signature scheme as initially described by Chaum [4]. The current specification has the card send its public key $\{K_C\}_{K_{CA}^{-1}}$ for the CA to include in the signed digital certificate. Using a blind signature scheme, the card would initially download a list of $\{\phi_i, exp\}$ pairs signed by the CA. The card would then create the proper certificate tuple $M = \{K_C, \{\phi_i, exp\}_{K_{CA}^{-1}}\}$. Given an RSA public key (K_C, n), the card would generate some random value r such that $gcd(r, n) = 1$, and send $M' = r^{K_{CA}} * M(\text{mod} n)$ to the CA. The real certificate, with the card's public key, is *blinded* by the random value r. The CA signs the returned value, $cert' = (M')^{K_{CA}^{-1}} = (r^{K_{CA}} * M)^{K_{CA}^{-1}}(\text{mod} n)$ and returns this blinded certificate. Therefore, the card can extract the proper signed certificate by merely computing $cert = cert' * r^{-1}(\text{mod} n)$.

Using a blind certificate scheme restricts the certificate authority from being able to log any information that would identify a user given its public key. If a single public key is used for all certificates, or even more than once for the same certificate, an adversary could still physically monitor the users of card readers to establish a key-identity relationship. To combat this anonymity attack, the user can establish a large number of one-time certificates, each using a different public-private key pair, from the CA. The CA would still be ignorant of the public keys used, and the card reader would see a different public key upon each access. Therefore, the user would achieve near total anonymity.

5 Conclusion

We have discussed an implementation of a smart card system to replace current technologies in use at MIT today. With concerns about privacy, our system uses methods that attempt to protect individual privacy while determining group memberships. With the goal of simple and distributed administration, the user/resource/policy management system allows a high degree of flexibility. These methods, however, introduce difficulties in revocation.

To maintain a secure environment and stop impersonation attacks, the system uses several authentication, encryption, and hashing technologies. Any future weakness found in these cryptosystems would endanger the security of our system.

Another problem of the system is a card reader's vulnerability to physical damage. So far, we have assumed the use and security of tamper-resistant hardware. An in-depth analysis of the different hardware protection options needs to be explored before actual implementation of this system.

Smart card technologies have the possibility of introducing many conveniences and applications to our daily routine. However, any future design must carefully weigh the tradeoff between providing functionality and protecting privacy. As the world becomes increasingly digital, this tradeoff becomes more apparent.

"The right to be left alone – the most comprehensive of rights, and the right most valued by a free people."

–Justice Louis Brandeis, Olmstead v. U.S. (1928)

References

[1] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM J. Computing.* 13(4):850-863. November 1984.

[2] Michael Burrows, Martin Abadi, and Roger Needham. A Logic of Authentication.
SRC Research Report 39. 1989.
http://gatekeeper.dec.com/pub/DEC/SRC/research-reports/abstracts/src-rr-039.html

[3] Graham Butler. Why Smart Cards? How will they be used in New Zealand? July 1996.
http://www.itaz.org.nz/pubs/smartcards/

[4] D. Chaum. Blind signatures for untraceable payments. In *Advances in Cryptology – Crypto '88.* SPringer-Verlag, 1988, pages 319-327.

[5] Alan O. Freier, Philip Karlton, and Paul C. Kocher. The SSL Protocol: Version 3.0.
Transport Layer Security Working Group, Internet-Draft. November 18, 1996.
http://home.netscape.com/eng/ssl3/draft302.txt

[6] John Kohl and B. Clifford Neuman. The Kerberos Network Authentication Service (Version 5).
RFC-1510. September 1993.
ftp://ftp.isi.edu/in-notes/rfc1510.txt

[7] Mondex. http://www.mondexusa.com/html/content/secur/security.htm

[8] News Release. Hitachi Launches H8/3112 Multos Chip. October 22, 1998.
http://www.hitachi.co.jp/New/cnews/E/1998/981022B.html

[9] NIST, FIPS PUB 180-1. Secure Hash Standard, April 1995.
http://csrc.nist.gov/fips/fip180-1.txt

[10] Scott Taschler. Securing Web Communications With SSL and Smart Cards. February 1998.
http://www.datakey.com/cardpage/web.htm

[11] Tom Wright, Information and Privacy Commissioner/ Ontario. Smart Cards. April, 1993.
http://www.ipc.on.ca/web_site.eng/matters/sum_pap/papers/smcard-e.htm

[12] UserLand Software. XML-RPC Specification.
http://www.xmlrpc.com/spec