ROUGH DRAFT
A Secure Successor to the MIT Card

Chris Laas

June 69, 2000

$Id: functions.tex,v 1.3 2000/08/16 13:48:59 golem Exp $

1 Overview

The majority of the design of this new system consists in writing down all the details we take for granted as cryptographers. This document serves to describe the high-level aspects of this design, and should be readable by a person with no background in cryptography or security. The second section aims to describe the various functions performed by the system, and in so doing, sets out the goals of the system, and puts limits on the abilities of the various parties in each transaction. The third section describes various aspects of the system which cut across individual transactions. The fourth section details the primary middle-level primitives (between the low-level cryptographic primitives and the high-level functions) which will be necessary to implement the functions, and lists the various properties that must be satisfied by these mechanisms. The fifth section draws the other sections together by describing, point by point, how each function is to be implemented using the mechanisms as primitives.

2 Function

The following is a list of actions that a user, Alice, could perform using her identification token; each example includes the interface seen by the user, an outline of the steps performed by the system, and the information recorded by each part of the system.

- Alice is on her way to building 66; the door opens to MIT community members. Alice couples her token to the door terminal; the terminal

tells her token "only MIT community members may pass", to which Alice's token replies with evidence that she is an MIT community member. The door opens. The door terminal will record the fact that someone passed, along with the time; as in every example, all of the technical aspects of the transaction should be recorded for security auditing purposes.

- Alice is outside room 26-100 to watch a showing of "The Matrix" at LSC. Students get a discount at LSC movies; also, tickets are sold in bundles at a discount (11 for the price of 10). When Alice presents her token to the LSC POS terminal, it demonstrates that she is a student (and thus entitled to a discount); depending on her settings, it may ask for confirmation or a PIN before giving LSC the $25 or so that the tickets cost. (The tickets might come in either a physical or an electronic form.) Alice's token may record a receipt for the transaction, and the LSC terminal stores its accounting and audit information (but not Alice's identity).

- From day to day, Alice makes many small purchases. She makes 17 five-cent copies of page 831 of her Introduction to Algorithms textbook; gets a 75-cent soda from a vending machine; and buys $21.54 worth of Twinkies from the corner grocery market, LaVerde's. In general, she does this by coupling her token to the POS terminal, and pressing the "Accept" button when it queries "$21.54 OK?". (If the purchase is large, or she has spent a large amount in a short time, her token may require a PIN as confirmation.) Receipt, accounting, and audit information will be available to both devices, but Alice's identity will not be divulged, and different purchases will be unlinkable (by computational means, at least).

  The primary properties of the small cash transactions will be those commonly known as unforgeability, no-framing, and untraceability and unlinkability; the first two take precedence, if there is a conflict, since the utmost concern of a cash system is to preserve the total quantity of money in the system. The system SHOULD have some degree of loss protection: if Alice loses her token, she should recover any portion which is not spent by a dishonest person who finds it. (PIN protection will benefit Alice here, of course.) The system SHOULD support multiple banks operating in the same currency market; indistinguishability of currency issued by different banks MAY be a feature. The system SHOULD also support multiple independent "currency markets"; for

example, LSC tickets might be represented by such a market, in which there existed only one payee.

The transaction system SHOULD have an offline fallback mode, although it may be assumed that, in general, network availability is plentiful. Ideally, normal operation would be a mix of online and offline operation, the proportion being set by policy and network conditions.

The features known as rip-spending (for greater fairness of transactions), transferability, and divisibility (of the Chaumian "cash check" variety) MAY be implemented. Of these, the first two deal largely with convenience issues, while the latter deals with a performance concern.

A variation: with an advanced phone system, the same mechanisms could be used to bill phone calls by the minute. However, since there is no point in providing unlinkability between the payment of one minute of a conversation to that payment of another minute of the same conversation, an optimized system to pay "phone ticks" MAY be implemented.[1]

- Along these same lines, Alice occasionally spends larger amounts of money, over a few hundred dollars or so: she buys textbooks and computer equipment, pays her rent, and mortgages her soul to pay tuition. Such transactions may not be as well served by a "pocket-cash" type system, and other, less anonymous, modes of transaction may be preferred. The system SHOULD be flexible in accomodating linkages with the existing banking system of debit, credit, and account management; it MAY implement some. Particularly useful systems might be:

  - A "Swiss bank" financial model using pseudonyms would allow linkage of purchases, creating an audit trail.
  - The ability to "anonymously" show a photo ID without creating a linkable trail could serve as evidence of ownership of a token.
  - Finally, the simplest and oldest method, authenticating oneself to existing banking systems using one's true identity, could be useful for debit-style and credit-style transactions.

- Athena user Alice wants to log in at one of MIT's pervasive computer clusters. She presents her token to a terminal attached to a worksta-

[1] See CAFE, e.g. paper in ESORICS '94, http://www.win.tue.nl/ berry/papers/esorics94.ps.gz

tion, which tells the token "I am workstation m56-129-12; authenticate yourself"; depending on policy, the token may require a PIN (and/or the workstation may require a password) to unlock the account. The token authenticates her to the Kerberos servers and the workstation starts her session. This method would improve the security of Athena, since password-only systems tend to be easily broken.

A variation would allow an anonymous login option to the Athena user community; this would be purely a policy decision by I/S.

The user token might also be used in certain secure operations over the network; for example, the token's participation might be required to obtain web browser certificates for a user.

- Policeman Bob asks to see some ID, please. Alice couples her token with the identification device held by Bob; Bob's device tells Alice's "I am Policeman Bob's device; identify yourself", providing a signed proof, which Alice's token may record. Depending on system policy and Alice's preference, Alice may have to enter a PIN to allow her token to identify her. When she does, her token presents Bob's device with a certificate (signed by the Card Office) containing at least her name, MIT ID number, photograph, and possibly other identifying information such as birthdate, gender, height, weight, eye color, etc. Bob's device has the option of recording this information, of course.

- The Campus Calvinball Club has an office on the fourth floor of W20, where they keep their expensive Calvinball equipment; nobody is allowed in the office who isn't a member of the group. The procedure to unlock the door is superficially similar to opening building 66; depending on the token-holder's preferences, confirmation might be required to prove membership in some groups. But, the members of the club agree that it should be impossible for anyone to tell (from the door terminal's records) who entered the office; Maintainer Mallory is a member of the club, but is also an administrator of the authorization system, and that knowledge would give him an unfair advantage in the game.

  Professor Alice and her aide, Graduate Student Bob, are the only people allowed in the super-secret Transmogrifier Technology Lab. Both of them are members of the CCC, so again, they'd prefer it if Mallory couldn't track their motions. But Transmogrifier equipment is expensive; if possible, the system MAY have a policy option which

allows some groups to have a designated authority, who can "open" the records, in case of a dispute.

In some situations, such as that of Employee Dave in the Cashier's Office, the system SHOULD allow policies in which Dave can only access the room at certain times of the day, such as during business hours.

- Alice may own a car; using the parking lot should be as easy as opening a door. The parking office has no need to know when certain people arrived and left; this should not be recorded. However, the parking office MAY wish to implement certain special policies, such as "no-passback".

- Student Alice lives at Fred the Dorm. Only Fred residents and their guests are permitted into the dorm, but Fred residents do not like the idea that their movements can be tracked. Just as in the student group example, there must be a group authorization mechanism; however, the changing guest lists mean that the access list needs to be able to be changed fairly frequently.

  Similar systems would be useful for library and sports facility access, and access to many other services.

- Alice occasionally borrows ice skates at Johnson Rink's Athletic Desk. Generally, she has to show some evidence that she's entitled to this service; but this authorization, by itself, is not enough, because the lender needs some assurance that the item will be returned at the appropriate time. Thus, the system SHOULD provide some mechanism to provide such assurance, probably in the form of some sort of collateral. (This issue MUST be dealt with somehow, because if it is ignored, the token itself will be used as collateral, breaking the security model.) In the case of ice skates, a deposit might be appropriate: a pre-agreed-upon sum of money could be transferred to the lender if the item is not returned in time.

- The library system is a special lender: a monetary deposit may not be appropriate collateral for a book. What's more, one does not merely wish to hide the identity of the borrower, but also, to some degree, the identity of the books being borrowed. The first issue may be addressed by the concept of an "identity collateral:" the borrowing procedure is anonymous, so that Mallory cannot determine which books Professor Alice is reading; however, if Alice fails to return a book, the borrowing

record is "opened" and the library determines her identity. (And metes out whatever punishment it deems necessary, hopefully giving her a warning first...) The second issue implies that, although users of the library system need to be able to determine whether a particular book is currently checked out, they (even the librarians) should not be able to contruct a list of books that are currently checked out, short of exhaustive search of the catalog; if possible, they should not be able to list the books that were checked out or in during a given time period, either. These goals might be achieved by careful application of anonymous database access protocols.

- If Alice locks herself out of her dormitory room, she needs to get lockout keys from front desk. Unlike the situation with a video rental, lockout keys are inherently non-anonymous; thus, to obtain them, Alice must simply present (and prove) a full ID. The scenario is very similar to the policeman scenario; however, Alice may have to confirm and/or type a PIN to unlock her identity information, depending on system policies and her preferences. Collateral (in the form of a cash deposit) might still be relevant, although another option would be to leave authorization to simply transfer funds directly from one account to another if Alice is late. (Another possibility would be to lock Alice out of certain services if she is late, although this is even trickier and definitely a policy issue.)

- In a more simple scenario, Alice works for MIT, and needs to pick up her paycheck. In order to do so, she needs to present a fully qualified ID to the payroll office; the scenario is like the previous, but there is no issue of collateral.

  In addition, a fully qualified ID may be used to view or modify official records, or to submit official forms, such as registration, financial aid, employment papers, and the like.

- The system could potentially keep medical information on one's person, easily accessible to paramedics with simple, fault-tolerant readers. The access policy regarding information such as allergies and medications must be carefully considered; the risk of injury due to failed authentication must be minimized, but medical information privacy is considered vital. One possible precautionary measure would place a "big red button" on the device to release the medical information of the owner, preventing remote attacks; at minimum, the device should record the release of information to deter attacks by POS terminals.

Whether or not to require cryptographic authorization in the form of a "medical technician" certificate is a tricky policy issue.

- Finally, Administrator Alice must maintain the system itself.

  Functions performed by the card office or other administrative bodies:

  - Production of new tokens.
  - Setting and changing of the system parameters and policies.
  - Auditing of logs for security purposes.
  - Data mining by authorized parties: for example, creation of image rosters (for classes, etc.) from the digital photos of every system user.

  Also, the membership of various groups (MIT Community, students / faculty / staff, departments, labs, classes, student activities, Physical Plant, etc.) is maintained by adding and removing members; this needs to be *decentralized* and accessible to any administrator of any such group (a broad class).

3 Details

Several aspects of the system cut across most of these interactions:

- Scaling: in fiscal year 1999, the Card Office processed 35,900 new MIT Cards. There are about 900 faculty, 10,000 students, and 10,000 staff at MIT.

- In many of the ID-showing and group-membership-showing protocols, it is left unspecified what sort of records are left to the verifier. If signed proofs (of knowledge of a secret key corresponding to the public key in a certificate) are used, the verifier's records can be later shown to any other party, who will be convinced that the transaction occurred. On the other hand, if a zero-knowledge proof mode is used, the verifier's records will not suffice to convince the third party, since the verifier could have forged those records. Several PKIs support both modes of operation. It's not yet clear which interactions require signed proofs and which require ZKPs; perhaps it is simply a policy issue (in which case both options SHOULD be offered to the system administrators).

- In a practical system, measures must be taken to facilitate the backup of records (receipts, protocol transcripts, and the like) to external systems, since tokens and terminals are unlikely to incorporate bulk storage, and they might be lost or destroyed in any case. However, care must be taken to ensure that this does not destroy the security properties of the system. It may be impractical for users to store backups on their own trusted, but flaky, hardware; on the other hand, backups stored in an "adversary"-controlled facility are vulnerable. (This is the case even if backups are stored in an encrypted form, since users inevitably choose poor passwords.)

- Secrecy of personal information is only one facet of data privacy; as important is the concept of data quality. Since data may be input incorrectly, or may change over time, there must be efficient mechanisms for changing records. This places some constraint on the level of decentralization in the system.

- On the opposite note, the system must support an adequate level of decentralization, especially in administrative functions, which, for organizational, efficiency, and security reasons, must be distributed over several administrative bodies. In addition, looking toward the future, the system should allow flexibility and interoperability among independent hardware implementations of e.g. terminals, so that separate administrative bodies may choose the appropriate embodiments of specialized functions. Finally, orthogonally to the need for decentralization of independent functions, the system should support multiple central authorities such as Banks, ID CAs, and the like.

4 Mechanism

4.1 Groups

Many of the functions described above fall into the general category of "group membership authorization" protocols; hence, a single underlying mechanism should be used to implement these transactions. To facilitate the creation of new applications of the system, more operations than the bare minimum to implement the above functions may be defined.

The group membership subsystem MUST support the following operations:

- Group creation and destruction

- Addition and removal of members
- Interactive proof of membership in a group (should not reveal any more about prover than membership in the group)

The subsystem MUST provide the following feature set:

- Additional restrictions attached to certificates: for example, time restrictions of validity.
- Decentralized group management: creation, destruction, and member addition and removal must be under the control of group owners.

The subsystem MAY support the following operations and features:

- Special modes of interactive proof of membership:
  - A **signature of knowledge** of the secret key corresponding to the public key of the group is a proof mode in which the verifier's view of the protocol *could not* be simulated, and thus serves as proof that a transaction occurred. An example: the verifier provides a random challenge, which the prover signs using her secret key.
  - A **zero-knowledge proof of knowledge** of the secret key corresponding to the public key of the group is a proof mode in which the verifier's view of the protocol *could* be simulated, and thus the verifier cannot prove to a third party that a transaction occurred. An example: an interactive zero-knowledge proof of knowledge of a discrete logarithm.
- Generation of a signature on a message (with the same privacy properties as proof of membership); verification of such signatures.
- If the group is tagged as openable, there may exist an "open" operation, in which a "group manager" (or other designated party) can open signatures (or possibly other transaction transcripts) to determine which group member made them. (Note: the cryptographic implementation of this is hairy.)

The privacy issue (security for the provers) turns out to be much thornier than it might at first seem. For example, a requirement for the system is that the proving protocol be private for the provers, to the greatest practical extent. Furthermore, a few systems, such as [OOK90], also provide

privacy of the membership lists of groups and the group lists of users; this comes at a distressing cost in manageability and security, one that seems insurmountable, and so this privacy concern is not a requirement for our system.[2] In all, there tends to be a great tangled tradeoff between security (for the verifiers), privacy, efficiency, and manageability. However, it seems possible to strike a balance in which all constraints are satisfied.

4.2 Individually-held certificates

Although individually-held certificates may be merely a special case of groups (which contain only one member), there are two differences which may merit independent implementation. First, different showings of the same certificate will always be linkable, and so for efficiency reasons not all the same privacy measures need to be taken as with group memberships. Second, an individual ID certificate may contain more specialized information (such as identifying features) than a group membership certificate. Some individually-held certificates may inherently be associated with a unique person (by means of a unique number, for example); others may be pseudonyms (e.g. bank accounts). (Care must be taken not to confuse pseudonymity with true anonymity in applications, however.)

The operations that MUST be supported are similar to the group case: there must be creation and revocation of certificates (although, as with destruction of groups or revocation of group membership, revocation of a certificate may not be explicitly via CRLs), and there must be a showing protocol to prove ownership of a certificate. As well, the optional features follow the lines of the group mechanism: the system MAY provide zero-knowledge and signature-of-knowledge interactive proof modes, and MAY support signatures on messages.

In the case of pseudonyms, a mechanism to keep one's "real" ID associated with the pseudonym in escrow could be implemented; however, this is tricky, and could severely limit privacy (depending on circumstances). It is worth considering.

4.3 Cash

A "pocket cash" account will be an important application of this system. While a very simple non-private centralized accounts-based implementation

[2]This is in large part due to the fact that we haven't been able to come up with a situation in which such privacy is clearly needed. Encountering such a situation might prompt us to rethink our system and try to work this privacy feature in.

can be constructed based on individually-held certificates, such a system cannot provide any privacy; hence, a privacy-protecting cash system should be constructed. Such systems are usually based on one-show tickets: whether they are called chits, tokens, cheques, coins, or tickets, if a spender attempts to show (spend) them twice, he is detected and/or prevented from doing so.

Despite its importance, the cash system must take second stage to the group authentication scheme. Hence, the simple non-private accounts-based cash system may be implemented as a placeholder until the group membership system is finished.

When the ticket-based cash system is implemented, the following properties (given by their usual names in the literature) will be considered:

- It MUST have *unforgeability* and *no double-spending*: in short, the amount of money in the system must be constant, and transfers must be intentional.
- It SHOULD have *untraceability* and *unlinkability* (of the "strong" variety described in [PSW95]): this is the basic privacy requirement.[3]
- It MUST have *no-framing*, which is typically only relevant in privacy-enabled systems.
- It SHOULD have *loss protection* (see [PW97]), but this is less important than privacy.
- It MAY have *rip-spending*; this is a neat idea, but its utility in practice is pretty questionable, and it adds some complexity. However, a form of rip-spending may be useful as a primitive for more high-level operations (for example, to ensure fairness in producing receipts).
- It MAY have *transferability*. Again, on paper this sounds like a neat idea (and even may seem vital), but in reality the system is expected to have a clear delineation between payers and payees, and to have a high degree of centralization and network availability. Hence, transferability should probably only be implemented if it is "easy" or if it is necessary for some other property (for example, for privacy in [Sim96]).
- It MAY implement *divisibility*. In some systems, such as [OO91], divisibility increases the complexity of the system hugely. However, it

[3] Pfitzmann and Waidner have a few rather interesting reads on electronic cash. They seem to have their heads screwed on straight.

may be possible to use one of the forms of "cash checks" described in [Cha89] and [CdBvH+89] to implement divisibility naturally, although these systems seem uniformly to be significantly more complex than "roughly equivalent" coin-based systems.

- It SHOULD support *multiple banks*. This is generally easy, and so should be included, although it's unlikely to be needed at the outset. It MAY have the property of *bank-untraceability*, i.e. payees cannot determine the which bank issued a piece of currency — however, this seems very hard.

- It SHOULD support offline operation, and SHOULD also operate in an online mode when network connectivity is available. This is a highly nontrivial issue: see section *XXX* for further discussion.

- It SHOULD produce receipts at all stages of operation. This will permit the various parties to prove claims such as "money was deposited to this account" or "this account has no money left in it."

Because electronic cash is inherently a complex system, the number of operations associated with the system is many. The parties involved in the system are: at least one *Bank*, a set of *payers*, a set of *payees*, and neutral third party judges when resolving disputes. The set of payers and the set of payees may be the same, although in this instance it seems likely that there will be a clear delineation between payers (students, faculty, staff, community, and other cardholders) and payees (cafeterias, merchants, vending machines, phones, and so on).

There will need to be a protocol for *establishing an account* with a bank (two, if payer and payee accounts are distinct, which seems likely). Of course, this protocol may be performed implicitly, at token initialization time, or via an out of band mechanism.

Of course, there will need to exist protocols for *withdrawal* of electronic money from an account by a payer; *payment* of electronic money from a payer to a payee; and *deposit* of electronic money to an account by a payee. These protocols are fairly self-explanatory. For online payees, the deposit protocol should take place immediately after and *as an integral part of* the payment protocol, so that fraudulent payments may be prevented as well as detected.

A whole slew of dispute resolution protocols must exist as well: for, if they are not implemented, the respective security properties are hollow and unenforcable. For example, there must exist a protocol for *proving fraud*

in which a bank can prove, to an honest third party, that a payer or payee has attempted to double-spend (if and only if that user did so, of course). There must exist a protocol for *proving withdrawal* so that, if a user claims that a bank has decreased his balance against his will, the bank can prove to an honest third party that the user requested a withdrawal of coins. For example, the bank could present a signed withdrawal request; if the user claims he never got the coins, it should be possible to re-issue them. Analogously, there should exist a protocol for *proving deposit*: a payee ought to be able to prove to an honest third party that he deposited coins in an account. If the bank claims never to have received the coins, it should be possible to re-send them. This looks easy, but it must be possible for anyone (not just the bank) to verify the validity of coins to be deposited.

An analogous protocol for *proving payment* would be nice, but depending on the system chosen, may prove to be difficult in practice. The goal would be that, if the payee claims not to have received coins from the payer, the payer would be able to show a proof to an honest third party that he paid; such proof might consist in protocol transcripts or receipts of some sort, although it would depend on the specific system. Another possibility would allow the payer to give the same coin to the payee again; however, this must not violate the untraceability requirement, and so care would need to be taken to use the same challenge. The dual to the proof of payment would be a protocol *proving receipt-of-goods*; this suffers from even worse difficulties than proofs of payment, since there is no electronic means of verifying receipt of goods. Perhaps something along the lines of what is done "in real life" can be implemented, however: the payer signs a receipt acknowledging receipt of goods before being permitted to take them away.

Finally, various miscellaneous protocols for supporting special features may be necessary. Loss tolerance, especially strong loss tolerance, requires a set of several additional protocols, including backup, reclaimation, settling, and dispute resolution protocols. Rip-spending requires changes to the payment protocol. Transferability often changes the character of a cash system significantly, and may involve the addition of a monetary transfer protocol. And, although divisibility usually does not change the character of all of the transaction types, it may, for example, add protocols for reclamation of "change".

4.4 Other Mechanisms

Although the primary functions of the system will be constructed using the above three mechanisms, some essentially low-level functionality will cut

across their boundaries. In particular, key management for the certificate authorities will be handled by a unified administrative body. Also, other system-wide administrative functions will need to be supported. For example, software updates will need to be performed periodically (although the "Smartcard Java Applets" model seems frightful, and the model used in this system should be considerably more conservative). Policy updates and other such information propagation will need to be handled routinely. In addition, compromise containment will need to be seriously addressed, and contingency plans (and mechanisms to carry out the plans) will need to be created.

5 Implementation of functions via mechanisms

5.1 Overview

For the most part, the high-level functions described in the first section can be fulfilled by using the middle-level mechanisms described above; some require some amount of additional "glue". Here, we fill in the details regarding this implementation.

5.2 Identity-based protocols

Several of the proposed card functions inherently require a unique identification of oneself as a specific individual. For these purposes, a simple application of individual certificates should suffice.

Every user token in the MIT system should contain at minimum two individually-held certificates. The first, most basic such certificate c_b should hold, as auxiliary information, only the holder's name and MIT ID number; this certificate's physical analogue is the information on the face of the current MIT card. The second, full, certificate c_f should hold all of the identifying information usually found on a driver's license, plus some more: name, MIT ID number, digital color photo, birthdate, gender, height, weight, eye color, and so on.

Whenever performing an authentication protocol of this sort, the two parties begin by establishing an encrypted channel using a standard technique (e.g. Diffie-Hellman); this prevents any identity information from accidentally leaking out to eavesdroppers. Man-in-the-middle attacks should be considered, but are usually impossible due to the direct physical connection between the communicating parties.[4]

[4]However, the Mafia fraud should be considered seriously. A diverse array of crypto-

The two IDs specified above could suffice for the following services:

- When logging in to Athena, the workstation would first identify itself to Alice's token (workstations would have individual "identities" under CAs independent from the "human-identifying" CAs); Alice would then use c_b to identify herself to the workstation (and the Kerberos KDCs). For users who wish to do so, it SHOULD be possible to use an independently issued ID certificate to log in to Athena; it SHOULD also be possible to use a separate ID certificate to log in to one's root or extra Kerberos instances.

- When identifying oneself to a police officer, the protocol is similar to that for logging in to Athena. Police officer Bob's terminal first authenticates itself to Alice by sending a signed message requesting identification; when Alice confirms the transaction, her token shows the certificate c_f to Bob using a signed showing protocol. Alice's token should record the transaction's relevant details.[5]

- To obtain lockout keys from a dormitory front desk, Alice presents her ID c_b, in much the same way as above. Alice should receive receipts for both the initial borrowing of the keys, and when she returns them.

- A showing protocol very similar to the above protocols can be used to pick up a paycheck or to view or modify official records or to submit official forms.

- If the system administrators and/or users choose to restrict access to medical information to authorized medical technicians, then such technicians would be required to perform a signed identity showing protocol in order to retrieve the medical information from a token. The signature would be kept as a receipt, proving that the information was released to that technician.

- Of course, it is worth noting that individual certificates are needed for several aspect of the group management protocols, including group creation and destruction, membership administration, and daily certificate issuing. Authentication will also be necessary for any system

graphic techniques exist to combat this fraud, but at least one should actually be implemented if the base system is susceptable to it.

[5]It would be more optimal if the nature of the transaction left Alice with a receipt, i.e. Alice receives a receipt if and only if Bob receives Alice's full certificate of identity. It's not clear that this is easily accomplished, however.

administration tasks for which access control is internal (i.e. not via an external system such as Kerberos).

5.3 Group-based protocols

Most of the privacy-enhanced functions described in the first section are implemented using the group subsystem as the primary mechanism. There will exist a few large, fairly static groups, and many smaller, more dynamic groups.

- At the topmost level (for the time being, at least) will lie the "MIT Community" group. Although initially all system users will be in this group, it has two very valid reasons for existence: the system should scale to include members of other communities, and the system must protect against those who have devices or device-simulating programs but do not have (and should not have) certificates. Members of the MIT Community group will have access to public buildings (e.g. doors to the Infinite Corridor such as those in building 66) and public resources such as the libraries. If IS sees fit, MIT Community might be allowed anonymous access to Athena services such as Web browsing.

- At the next level will be the broadest division, of community members into "students", "faculty", "staff", "other community", and so on. As an example, "student" status might entitle a user to various discounts (e.g. the LSC example).

- Beyond this point is a mix of various groups of various sizes: departments, labs, research groups, offices, students in a class, student activities, living groups, guest lists, physical plant workers, and so on. In general, each may be a normal group; auxiliary information may be used to specify access restrictions (such as certain times of the day).

  The mechanisms, as given, do not provide a way to "open" records as described in the "Transmogrifier Technology Lab" example. Such a mechanism could be created using the techniques given in [CS97], although they are extremely complex and inefficient, and so are not suited for the usual case.

- The parking office may use the group mechanism to regulate access to the parking lots, which is perfectly reasonable. The parking office may also wish to implement the policy "no-passback": once a car has entered the lot, it should not be possible for the same token to enter

the lot again before he exits. This strategy aims to reduce over-parking due to lending of access tokens.

No-passback can be implemented simply in an anonymous manner. In the simplest form, a parking lot user could be initially endowed with a "parking chit", simply a random value in a large space; when entering the lot, he presents the chit along with his authorization to enter (and is denied access if the chit is not found in the database of issued chits). Once a chit has been used, it cannot be reused to enter the lot; however, upon exiting the lot, the user is given a new chit which he can use to enter later on.

This mechanism does not achieve full anonymity, because the chit issued to a user when he leaves is the same as that which he presents in the morning when he arrives, and hence the transactions are linkable. This problem can be easily solved using blind signatures, however. The chit, when issued, is a blind signature issued by the parking authority; it can only be "deposited" once, but the withdrawal and deposit transactions are unlinkable. (As should be clear from this, this chit system could use the same one-show tickets as a mechanism as does the cash system.)

Finally, a chit system might be prone to becoming out-of-sync, if a user ever loses his chit, or the parking authority database ever crashes. It might be more practical to re-issue chits every morning, on the assumption that cheaters who stay in the parking lot overnight are negligible.

5.4 One-show-ticket-based protocols

The primary purpose of the one-show-ticket mechanism is as a financial transaction subsystem; however, its protocols may well prove useful as parts of other applications. Since it is more specialized than the other mechanisms, the financially-relevant features are discussed in the mechanisms section, above. In general, a cash transaction will begin with a vendor terminal sending a signed request for a certain cash amount (5 cents, 75 cents, $21.54) to the user token, simultaneously identifying itself; the token will ask for confirmation from the user and/or a PIN from the user (steps which may be omitted if the user has requested that small transactions go through without confirmation and/or a PIN); and, if confirmed, the token will transmit the electronic cash to the vendor. If the vendor is online (as most will be), he will verify the validity of the coins and approve the transaction. In certain

systems, the vendor will return "change". Preferably, the vendor should return a receipt to the payer as well, but it's not clear how to ensure that the vendor is forced to do so.

The ticket system might also be used in some way in a "phone ticks" system, but in actuality, if such a system is implemented, it will probably be a semi-independent system using hash sticks or the like.

The one-show-ticket subsystem can be dropped in directly to provide electronic LSC tickets. The full flexibility of electronic cash is overkill for simple LSC tickets, but it does not hurt.

As noted above, simple one-show tickets can be used as parking chits in the no-passback system described above.

The identity escrow subsystem of the library subsystem may be poorly implemented with n-show tickets; however, such a mechanism would be very inefficient, and probably not worth it unless for demonstration purposes.

5.5 Special protocols

As noted in the "functions" section, some forms of larger financial transactions may be supported in the future. These are currently vaporware, which is perfectly OK, since they are not high priorities.

One may wish to escrow digital cash as collateral for ice skates and the like. Verifiable signature sharing ([FR95], [Bur96]) provides a perfectly reasonable answer here; it's not overly complex, and is designed for exactly this purpose. It may be a little while before we get around to implementing it, though, as it's not the most pressing issue.

The library system protocol, as described in the "functions" section, is extremely challenging: there exists no good answer yet, and it is an open research problem. If a good solution shows up, we will use it; otherwise, we will make do with the mechanisms described above. Library access can be easily controlled in a non-anonymous way by using ID certificates (or pseudonymous "library card" ID certficiates), and it may be possible to make improvements using trusted third parties, as well.

References

[Bur96] Mike Burmester. Homomorphisms of secret sharing schemes: A tool for verifiable signature sharing. In Ueli Maurer, editor, *Advances in Cryptology—EUROCRYPT 96*, volume 1070 of *Lecture Notes in Computer Science*, pages 96–106. Springer-Verlag, 12–16 May 1996.

[CdBvH+89] David Chaum, Bert den Boer, Eugène van Heyst, Stig Mjølsnes, and Adri Steenbeek. Efficient offline electronic checks (extended abstract). In Quisquater and Vandewalle [QV89], pages 294–301.

[Cha89] David Chaum. Online cash checks. In Quisquater and Vandewalle [QV89], pages 288–293.

[CS97] Jan Camenisch and Markus Stadler. Efficient group signature schemes for large groups (extended abstract). In Burton S. Kaliski Jr., editor, *Advances in Cryptology—CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 410–424. Springer-Verlag, 17–21 August 1997.

[FR95] Matthew K. Franklin and Michael K. Reiter. Verifiable signature sharing. In Guillou and Quisquater [GQ95], pages 50–63.

[GQ95] Louis C. Guillou and Jean-Jacques Quisquater, editors. *Advances in Cryptology—EUROCRYPT 95*, volume 921 of *Lecture Notes in Computer Science*. Springer-Verlag, 21–25 May 1995.

[OO91] Tatsuaki Okamoto and Kazuo Ohta. Universal electronic cash. In J. Feigenbaum, editor, *Advances in Cryptology—CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 324–337. Springer-Verlag, 1992, 11–15 August 1991.

[OOK90] Kazuo Ohta, Tatsuaki Okamoto, and Kenji Koyama. Membership authentication for hierarchical multigroups using the extended Fiat-Shamir scheme. In I. B. Damgård, editor, *Advances in Cryptology—EUROCRYPT 90*, volume 473 of *Lecture Notes in Computer Science*, pages 446–457. Springer-Verlag, 1991, 21–24 May 1990.

[PSW95] Birgit Pfitzmann, Matthias Schunter, and Michael Waidner. How to break another "provably secure" payment system. In Guillou and Quisquater [GQ95], pages 121–132.

[PW97] Birgit Pfitzmann and Michael Waidner. Strong loss tolerance of electronic coin systems. *ACM Transactions on Computer Systems*, 15(1):194–213, February 1997.

[QV89] J.-J. Quisquater and J. Vandewalle, editors. *Advances in Cryptology—EUROCRYPT 89*, volume 434 of *Lecture Notes in Computer Science*. Springer-Verlag, 1990, 10–13 April 1989.

[Sim96] Daniel R. Simon. Anonymous communication and anonymous cash. In Neal Koblitz, editor, *Advances in Cryptology—CRYPTO '96*, volume 1109 of *Lecture Notes in Computer Science*, pages 61–73. Springer-Verlag, 18–22 August 1996.