A Platform for Privately Defined Currencies, Loyalty Credits, and Play Money

David P. Maher

AT&T Labs Research
Room B245
180 Park Avenue
Florham Park, NJ 07932-0971
dpm@research.att.com

Abstract. We use techniques from financial cryptography to define new electronic currencies that are suitable for many applications. We use a platform approach to allow a single, world-wide infrastructure to support a practically unlimited number of new currencies. The platform permits new currencies to be defined with little effort, and allows an individual to effectively manage and use perhaps a few dozen of those currencies that he finds personally useful. We describe the structures and mechanisms of the platform, various applications, and the risks associated with its use.

1 Introduction

We have designed a system that allows individuals to use an efficient, publicly available infrastructure to define and dispense private currencies and other types of value such as loyalty credits and play money. We want to make it straightforward for a corner sandwich shop owner to set up a loyalty system or for an airline to enhance its frequent flyer program. We want to allow a corporation to easily set up a private scrip system to simplify internal account transfers, and to make it easy for children to issue and exchange play money for games they might play over a network. Likewise, we want to make it very easy for people to use multiple currencies and value systems by using a single simple device. The platform, which uses smart cards, could in the future allow new currencies to be instantly defined with as little effort as it takes to visit a web site and complete a form. The platform is extremely scaleable, easily supporting millions of currencies, and allows individuals to practically use as many currencies or value systems as one can imagine. In the long run, we believe that the marginal cost of defining and supporting a new currency that can be used world wide can approach zero.

An interesting aspect of the system is that it can be used to conveniently exchange value of different types, depending on their rules of use. Thus, an individual could exchange frequent flier points for free haircuts or meals. We briefly discuss the implications of such possibilities, and claim that firm conclusions regarding effects on various micro-economies can be obtained only through experimentation. Fortunately, the platform described here should make it easy to experiment with various rules of value exchange.

The platform that enables this free use of new currencies is based on a cryptographic key management system that was designed to be simple yet very effective. The art here is in finding a minimal system that is broadly useful.

Before discussing some of the technical details, we explain how someone might use the system. Then we describe the platform components, how they work, some of the security risks, and some of the risk mitigation possibilities. Then we return to a discussion of applications of the platform.

2 Definition

We define *currency* as any medium of exchange. Currency is not necessarily legal tender, and in the context of this paper, it usually is not. The platform that we describe allows a person to define either *directed* or *undirected* systems of currency. In directed systems, currency follows a directed path among classes of users and is not normally *negotiable* within the system. In undirected systems, currency follows undirected paths and the currency is negotiable by peers in the system much as US Federal Reserve Notes are today.

3 End-user's view of the platform

We suppose that smart cards with certain properties defined below are universally distributed and accessible to anyone. For simplicity, assume that there is one distributor of smart cards that are called Xcards who maintains a web-site where new forms of currency that can be carried on the card may be defined. Let's say that a grocery store chain wants to set up a loyalty point system. A representative who will originate the points visits the card distributor's web-site and completes a form that includes

- a name for the currency (e.g. "Shop and Save Green Points")
- properties (such as whether the point system is directed or undirected)
- An initial value sum to be distributed to this originator

At the web-site, a script is executed whereby the entries on the form are checked for consistency, and the name given to the points is ensured to be unique. If all goes well, a multi-layer secure protocol is executed between the web-site and the grocery chain representative's client whereby the representative's smart card is initialized with the specified number of Green points.

Now, the grocery chain can use electronic means to distribute the Green Points internally for further distribution at grocery store point-of-sale terminals. Points are distributed from the originator's Xcard to intermediate Xcards held by store managers, and then to Xcards at Point-Of-Sale terminals. When a consumer pays for groceries at such a terminal, he can insert his Xcard and receive a certain number of points through a simple value transfer protocol. If the consumer is receiving Shop and Save Green Points for the first time, the consumer’s Xcard is automatically initialized, without disturbing any of the other currencies this consumer may be using. If Green Points constitute a directed currency system, the consumer can redeem the points only at a Shop and Save grocery. If the system is undirected, the consumer can exchange the points with any other holder of an Xcard, in person or over the Internet, for any value he can negotiate. These Green Points have some initial value as soon as Shop and Save declares a redemption schedule which may consist of prizes, discounts, or any number of other goods or services. The Grocery chain might easily specify that Green Points could be officially exchanged for a variety of other types of currency such as frequent flyer points or credits for compact disks, or whatever.

4 System Goals

We endeavor to make the platform as simple as possible, with as little overhead as possible. We want currencies to:

1. *Be very easy to define:* We want the platform to allow an infrastructure to be shared by perhaps millions of merchants and other individuals worldwide who may want to define a value system that captures everything from loyalty to discounts to hypothetical or imaginary value in instructional or recreational games.
2. *Be very easy to use:* Many people get dozens of opportunities to participate in various currency and loyalty systems. However, they are often a hassle to use, people need to carry around various cards that often get lost, and people often forget that they are participating.
3. *Have a simple, low overhead infrastructure:* We would like a certain uniformity of procedure and a sharing of costs. We would like the marginal cost of supporting a new currency to be negligible. From the point of view of the user, one or two cards may allow participation in a large number of currency schemes in a number of domains, including mercantile, corporate and recreational domains.
4. *Have enough flexibility to allow schemes to be suited to a number of purposes and environments*: We want to allow currency systems to interact, and strengthen one another, and to provide a viable means of trade that can operate at grass-roots levels.

5 The platform components

The platform consists of a number of specifications that we describe here at a very high level. The tangible components are smart cards (or some other trusted computing base), their associated interface devices, and pages on card issuer web-sites. A full set of detailed specifications has not yet been completed. Omitted here is a description of the card issuer’s Key Management System. However, one may infer the functional requirements of the Key Management System from the descriptions of the other parts of the platform. The intangible elements of the platform consist of various data structures, cryptographic schema, process descriptions, and interface specifications.

5.1 Smart cards

For currency management, we suppose the availability of single chip smart cards (as specified in ISO Standard 7816). In principle, other types of devices can substitute for smart cards. In general, we need something that is portable and has some computing capability with some tamper resistance. Smart Cards are the paradigm that we have chosen, although in reality they typically do not presently have a great deal of tamper resistance. We believe that today, for many applications, their security capabilities will suffice, and there are prospects for improvements to their security design in the future.

We presume that each smart card chip that complies with the platform specification has the following:

- A processor that supports cryptographic functions
- Permanent program memory (ROM)
- A small amount of protected nonvolatile memory (PNVM) that is used for chip-specific cryptographic keys
- A card unique secret symmetric key (CSK) stored in PNVM
- A unique ID vector
- Non-volatile RAM (NVM) for additional programs and data
- A number of card issuer's public keys (CIPKs)
- A random number generator
- A card-unique asymmetric (public, private) key pair
- A certificate for the public key signed by the card issuer

During the final stages of manufacturing, a program on the chip generates the CSK for use with symmetric ciphers. This key never leaves the chip but is stored in protected non-volatile memory. The chip[1] also generates a private and public key pair for an asymmetric cipher system such as RSA. The private key is stored, encrypted, in non-volatile memory and the public key and chip ID are exported for certification by the manufacturer. The chip then receives and stores a unique chip certificate for use with the asymmetric key pair, signed by the issuer using one of the RSA secret keys whose public counterpart is stored in the card as a CIPK. In general we presume that the PNVM resists tampering, and that the CSK stored there is used to ensure the integrity and confidentiality of the rest of the key information that is stored in regular NVM.

The set of CIPKs on the chip includes at least two RSA public keys. One is used for signing card credentials, and one is used for signing messages generated in the currency definition process described below. The card may also include a Diffie-

[1] For chips with weak cryptographic function capabilities, it may be more reasonable to simply inject the keys and the accompanying certificate.

Hellman public key for use in generating symmetric keys that are shared between the card issuer and the card to support some optional data backup protocols.

5.2 Card Interface Devices

Users and networks interact with smart cards through interface devices. Consumers will use two types of devices: The first is a hand-held device resembling a calculator with a small keypad and display and one or two slots slot in which cards are inserted. This device will usually have its own processor, and may have non-volatile memory that stores currency information that can be managed by the user in conjunction with the capabilities of the card. Such a device can inexpensively include enough memory to support hundreds of currencies. With an Interface Device that has two slots, value can be exchanged between two cards using just that device. The second type of interface device is a PC or other computer terminal that includes a smart card reader. This device may be connected to a network. The non-volatile memory in the PC as well as in the network can store currency information that can be managed by an individual using her card. For both types of interface devices, the keyboard and display support the command interface between the user and the card. We expect standards to emerge that will support the definition of the interfaces between cards and interface devices and the processes that run on those devices. Microsoft, for example has proposed such a standard, called PCSC, for use with their operating systems.

Merchants can use similar interface devices, but they will be optimized for point of sale or network applications.

5.3 Card Issuer Web Sites

We speculate that major banks, telecommunications companies, and other organizations will be issuing smart cards to their customers. Although we do not believe that a given consumer will maintain all of their applications on one card, we do believe that multiple applications will be bundled on cards. Multiple applications can be used to increase the likelihood that a consumer will carry and use a given card. So, it will be in the card issuer’s interest to maintain a web-site that will support the use of their cards. Pages on the site will be used to allow the definition of new currencies that can be available for use by anyone. Pages will contain forms that can be completed by anyone wishing to define a new currency. Java programs interact with the user’s card, and CGI scripts interact with card issuer’s databases. The process of defining a new currency is discussed below.

We also expect that the card-issuer web-site will provide customer support for cards. In particular, a card memory backup and error recovery service can be provided.

5.4 Currency Trees

Programs on the smart card chip can construct binary signature trees similar to those suggested by Merkle [1]. We call these currency trees. Here, each leaf of the tree will contain data for a particular currency, as described below. Interior nodes will contain the result of the keyed CBC MAC [2] of the data in its two child nodes. The root of the tree will be stored in the chip card's non-volatile memory. Typically the key used for this MAC will be the smart card chip's own secret key or CSK. In some cases, it will be a key shared with the card issuer by use of one of the issuer's public Diffie-Hellman keys (CIPKs) stored in the chip.

In general, programs on the chip card will manipulate a number of currency trees. Note that the chip need only store the root value of a given tree. The details of some currencies that are most frequently used will be kept in a currency tree stored on the chip itself. The details of other currencies that are less often used are kept in trees stored in smart card interface devices the size of small calculators, or on Personal Computers or network appliances.

The reader may note that a non-keyed hash function could be used here in place of the keyed MAC, since we are storing the root of the hash tree in a secured place on the smart card chip. However, we believe that the keyed MAC can be more efficient, and supports some features that we will discuss later.

5.5 Currency tree entries

A currency tree entry is a record that contains a small amount of information:

1. Name of the currency
2. Currency type
3. Role played by this user (originator, redeemer, consumer, etc, see below)
4. Transaction sequence number
5. An integer amount corresponding to the value in this currency owned by the bearer
6. Special permissions

The currency name must be globally unique. The currency type is a designator that determines the rules of use of the currency. For example is currency flow undirected, allowing full negotiability, or directed, requiring the designation of cards that have special permissions allowing the redemption of currency? In this latter case, it is necessary to store the designated role in the currency tree record. So, a user may play the role of originator of new currency, or redeemer, whereby the user in a directed system can accept payment using the currency. The user's transaction serial number is required to defend against replays of messages used to transfer value. Special permissions are required to mint more currency, or to designate user-roles.

5.6 Card API.

There is a card API that describes the formal command set that is used to interact with the card currency application. This API allows applications running on various interface devices such as PCs, workstations, and personal electronic wallets to interact with the currency applications on the card.

5.7 Processes

We describe the most fundamental processes supported by the platform at a very high level. We have endeavored to minimize the complexity of the processes. We observe, however, that as we attempt to mitigate risks, provide for robust recovery from errors, and allow for renewability of security schemes and parameters, the complexity of the system grows. For the moment we favor a minimal approach until we can determine what additional processes are truly useful and clearly pay their way in exchange for the added complexity.

Defining currencies and originating value. As mentioned above, the card issuer will maintain a web-site that will include scripts that allow the definition of new currencies in an automated fashion. Any visitor to the site can define a new currency. There is no need to identify the visitor. A Java program can carry out the currency definition protocol on the visitor's client. The visitor completes a form, supplying the currency name (which is checked at the server for uniqueness), and then the currency type value (directed or undirected) and the initial amount to be originated are entered. The visitor will also specify whether he, as the originator, wants the privilege of generating more of the same currency in the future, without interacting with the issuer's server. The server computes a new currency data structure, signs it using the secret key associated with one of the RSA CIPKs stored in the visitor's card, and sends it to the visitor's client. The client issues a card API command to accept the new currency structure. The visitor's card verifies the authenticity of the currency structure using the CIPK, and then adds the new currency to its local currency tree using the MAC key (usually the card's CSK) associated with that tree. At this point, the new currency is defined, and an initial amount has been minted.

The integrity of this process is highly dependent on the uniqueness of the currency name, and the integrity of the CIPK. If an impostor who knows just the CIPK public key carries out this protocol with an issuer, a new currency may still be defined, however, the currency will not be useable, as the value transfer protocol described below requires use of the issuer certified asymmetric key pair. To prevent impostors from using this service, we may want to require the visitor card to use its certified cryptographic keys, binding them into the currency definition protocol.

Creating more value. The originator of a currency may, if the currency rules allow, create more value. A simple request is sent to the originator's card, naming the currency and the new amount to be put on that card. The card complies if the data in the currency tree affirms the privilege to create more value. Note, below it is stated that the originator may grant currency creation privileges to others. Cards with this ability need to be carefully managed. In many circumstances, this privilege is not needed or can be limited to a very few cards.

Transferring value. Currency value can be sent from one card to another by use of a value transfer protocol such as one of those specified in CEN standard 1546, part 2 [3]. In these protocols, payer and payee cards first exchange transaction details including the certificates of their respective cards and the amount to be transferred. Recall these certificates bind a card ID to a public key. In CEN 1546, the payer and payee exchange signed messages to complete the transfer.

In our case, we follow a similar arrangement, except that we need to deal with selection of currency. In the initialization phase, the name of the currency to be used is included in the transaction preamble, when the card certificates are exchanged. In our protocol, it is not necessary for the payee card to initially recognize the currency name, as the card may not be initialized with that currency. After transaction initialization, the payee's card then sends a formal signed message, requesting payment. Many of the payment details are repeated in this message, including the currency name, amount, and transaction sequence numbers used to foil replay attacks. The payer then debits the currency amount, recomputes the currency tree values from this leaf to the root, stores the new root value, and sends a signed message to the payee asserting that the debit has been made. The payee, upon receiving the message and verifying its authenticity, increments its currency value amount. If the currency name is unknown to the payee's card, a new leaf is automatically created for it in a currency tree. In either case, tree values are then recomputed along the path from this leaf to the root, and the new root is stored on the card. In some variations of the protocol, the payee then sends a signed acknowledgement message back to the payer.

Of course, the actual payee and payer are oblivious to the details carried out by their cards. They do interact with their cards through an interface device that typically has a small display and keypad. For value transfer, the payer need only specify the amount to be paid (though it is the payee's card that formally requests it), and the payee signifies acceptance.

Granting special permissions. Special permissions that can be recorded in a currency leaf's data structure include:

- bearer is an originator (and can therefore grant special permissions in a directed system)
- bearer can mint more currency
- bearer can grant permission to mint more currency
- bearer may dispense this currency to consumers in a directed system
- bearer is authorized to redeem value in a directed system
- bearer can grant special permissions in a directed system

Special permissions are necessary in a directed currency system, as the currency is non-negotiable by consumers, but there needs to be special classes of cards that can dispense the currency and redeem it. An originator can set up those classes by using the permission granting protocol.

For a given currency, a user may have special permissions. Those permissions are exercised through a protocol that involves commands in the API. Permissions are granted using signed messages between grantee and grantor, similar to the value transfer protocol described above.

Whether the system is directed or not, we may want to increase the amount of currency in circulation. This could be done with the cooperation of a card issuer (by reserving the right for the currency originator to go back to the card issuer to create more currency, or it can be done with cards specifically designated to increase the currency supply.

Maintaining currency trees. Merkle hash trees are ideal for this platform, as they are very efficient when there is a relatively large amount of data that needs to be selectively verified and updated. The use of a symmetric key MAC makes transactions even more efficient. Very little data needs to be stored within the smart card, and only a small number of MAC operations (log n, where n is the number of currencies in the tree) need to be called.

We believe that once alternate currencies are made easy to use, then people will use dozens of currencies from different loyalty programs, corporate scrip, and other applications.

The trees maintain the integrity of the various currency amounts, and permit the backup and off-loading of the details from each user's card. We envision currency systems to be just one of many applications that may be included on a given smart card. There is usually very little room for applications on a card in any case, as current cards typically have very little memory, perhaps 16K bytes of ROM and 2K to 8K bytes of NVRAM. We want to minimize the latter, as the amount of NVRAM significantly contributes to the expense of the card. Thus, having a single platform for multiple currencies is very useful, and having the ability to store details of the applications off-chip with integrity is another bonus. We also want to have provisions in case cards are lost, since we are concentrating a lot of value. In some cases, the amount of value can be real and significant such as in the application of a corporate scrip system.

A user's card interface device (hand-held device or PC or workstation) can maintain card currency trees and even archive them redundantly on public servers. The card only need keep the value of the root of the tree in the card chip's memory. The interface device can run programs to exchange segments of trees that are stored on-chip with trees that are stored off-chip. The card can always verify tree integrity and recalculate tree values when trees are reconfigured in order to allow often used currencies to be stored on-chip, and less-often used currencies to be stored in trees on the PC. Copies of trees can be stored off-chip including the transaction sequence numbers included in the authenticated data. When a card is lost or becomes corrupted, one can appeal to the archived versions. One method for verifying integrity includes a simple escrow method using the card issuer. If the tree is stored with a MAC key and a backup vector constructed from the card issuer's public key stored on the chip, then the escrow method described in [4] can be used to allow the issuer to verify the integrity of the given currency tree. Of course, an unscrupulous user may execute several transactions in his favor after having claimed to have lost his card. This is risky in directed currency systems where the redeemer of currency may have records that can be reconciled with the values in the old trees. A currency transaction serial number could be a give-away. Even in the case of negotiable currencies, the archived values can be useful in decisions as to whether to restore value.

6 Risk Analysis

A number of risks must be analyzed with this platform. In general, risks need to be analyzed in the context of a specific application. However, there are common aspects of the platform that will impute risks across all applications. The most glaring risks for each currency involve dependence on a third party, namely the card issuer who is in a position to mint any currency. Such risks can easily be mitigated to an extent, and the risks will obviously be acceptable in some cases, but will clearly not be acceptable in others. One could use this technology in stock exchanges, where dependence on another third party would be a major issue. Specific risk mitigation for such cases can be added to the platform whereby the role of the card issuer in the definition of the currency can be diluted through redundancy techniques as described in Mike Reiter's Rampart System [5], but that is beyond the scope of this paper.

A second obvious class of risks has to do with the integrity of smart cards. We implicitly depend on the assumption that it is non-trivial to make modifications to the card that will cause keys to be revealed, and computations to be in error. This set of risks is mainly beyond the scope of this paper, however a reference that attempts to put some of these risks in perspective can be found in [6].

The risk of using a common platform is itself interesting, as diversity of approach will in general reduce risk. However, there are some advantages to the common platform in that the cost of risk mitigation can be spread across many applications. A number of diversification methods can be used within the platform itself. In general, we have favored the risk reduction approach of simplicity, while allowing the additional complexity of some mitigation measures to be applied when the benefits can be more clearly accrued.

6.1 Risks associated with the card issuer

The card issuer has the responsibility for providing cards with system credentials, signing those credentials on a per card basis. The card issuer manages the secret key(s) that correspond to the CIPKs that are stored in the cards. We advocate using different keys for signing credentials from keys used to sign messages that initiate currencies. The credential keys typically need to be managed at the point of card origination, while the message signing keys need to be used at a web-site. If either type of key is compromised, however, the scheme is severely affected.

When a credential-signing key is compromised, that key along with other information can be used to create a practically unlimited number of counterfeit card emulators running on personal computers. Such emulators can produce arbitrarily large currency value sums and can behave outwardly in a way that is indistinguishable from an authentic card, except in ways that may be evident only to the user. For example, emulators can be used to initiate currency and to exchange value with legitimate cards.

When a currency initiation message-signing key is compromised, counterfeit currency that is indistinguishable from legitimate currency can be injected into the supply. Such events can be just as devastating as the production of counterfeit card emulators. Thus, we need some effective risk mitigation measures.

Instead of using traditional certificate revocation measures to mitigate the risks associated with possible key compromise, and depending on protocols for replaacing compromised keys, we favor an approach that is better matched to the mixed on-line and off-line use that we expect of the cards. In particular, we generate several extra CIPKs and store them on each card. The extra private keys are stored in a highly secure place (perhaps split among several places) until they need to be used. Individual CIPKs can be activated and de-activated using an authenticated system message that acts like a virus that is passed from card to card whenever two cards interact. The virus can be seeded from the issuer's web-site, as well as through merchants. Known on-line users can be actively contacted. Only a few users will need to be called, as the virus will propagate efficiently if it is transferred during every transaction. The card lifecycle needs to be taken into account here. We expect a given card to be in the field for 2 to 4 years. New cards can include new keys that get activated as old cards are retired.

6.2 Risks associated with card integrity

If the tamper resistance mechanisms of a given card fail in certain ways, then the card may give up its secrets, providing the possibility that clones of that card may be created. Such clones may not be expected to follow rules of use, and can be allowed to have forge arbitrarily large currency sums. There are a number of ways to violate the physical and logical integrity of the card. Classes of attacks include bus probing, memory imaging, fault induction, and the observation of channels that leak cryptographic information. Examples of the latter include timing attacks and observation of dynamic microprocessor characteristics such RF emissions and power consumption. A complete description of these is beyond the scope of this paper.

Although CIPKs are "public keys", in this system, there is no need to make them public. They could just as well be kept secret. However, in the case where they are revealed, there are more possibilities for a cryptographic attack. Therefore, appropriate key sizes must be chosen. However, for keys used during value transfer, there is a trade-off with transaction times. For mass transit payment systems, toll roads, and the like, the transaction time is severely constrained, and we will want to minimize the time required for all operations including public key cryptographic operations.

6.3 Risks associated with currency management

Currency originators bear a responsibility for managing the currency supply and the currency flow. There are a number of risks associated with mismanagement. In general, the currency management risks are independent from one currency to another, in contrast to the system level risks associated with key management. In our platform, originators of currency can authorize new currency issues, and grant privileges to people to dispense or to issue more of a given currency. There is a risk that these privileges can be abused.

6.4 Risks associated with user behavior

A major risk is loss of a card. There are a number of data recovery techniques that are reasonable to use in the case when the card is used exclusively on-line. For example, we can implement a service and a protocol that backs up the card information and currency trees to the issuer’s web site, and locks the card so that it cannot be used. The card issuer site knows the lock state, and if the card is lost when in the locked state, a new card can be issued, with the card currency trees fully restored. To exit the locked state in order to use the card, the user would have to participate in a simple unlocking protocol with the issuer’s site. These protocols can be very straightforward, and the user interface would be a simple toggle switch on the on-line interface device GUI. Cards that are used off-line, or are lost while in the unlocked state present a much harder problem. A backup process can still be used, but the backup data must be presumed to be incomplete. In general, we do not know who the currency originators are, but individual originators could register with the card issuer, and provide a policy for dealing with lost cards.

6.5 Risk mitigation concerns

In considering risk mitigation measures, there are tradeoffs to be made among challenges to simplicity, cost of risk mitigation, and the comfort obtained from the mitigation schemes. We will claim that for most applications the simplest approach, consistent with what has been described here will suffice, and that over time, as the infrastructure scales up, the cost of more elaborate procedures to protect keys will fall within the range of reason. The system design is modular enough that strengthening the security of certain procedures will often not require massive system redesign.

7 Applications and Negotiability

We have mentioned the number of loyalty programs that seem to spring up everywhere. We believe that this platform can make such programs much easier to use, and that we can thereby enhance the value of these programs. With a common platform, we can make it much more feasible for people to trade value of different types. This itself can enhance the value of a given loyalty program. Frequent Flyer Points that are highly negotiable can be much more valuable, and can therefore be more effective in promoting loyalty. Of course, there are downsides to negotiability to the airline. It would be interesting to see how the tradeoffs evolve. There are some psychological aspects of these currencies that need to be better understood. Will people value them against a single standard such as some legal tender? Or will people continue to put these currencies in a special class? One could always transfer undirected currency value in return for cash, and one might imagine a broker system arising to aid this. Loyalty programs are supported by two main economic principles: First, the consumer of loyalty points values the points more than the redeemer or issuer. Second, the consumer is reinforced for the use of the issuer’s products or services. Loyalty points that are negotiable still operate within these principles.

One can imagine two-way transfers that involve different currencies, such as in this anecdote: Parent: “Johnny! You mean that you traded your school good-behavior points for some ‘Quake IV’ ammunition credits?” Johnny: “Ted was grounded until he got enough points at school, so I got a good deal.” Johnny, who may not have cared about the good-behavior points before, now finds value in them, as he can impress his friends with his prowess at a popular Internet game. He is now motivated to collect more points. Of course, Ted is another story. This example illustrates the idea that value is variable and can be enhanced by opportunity. We often don’t capture this in most alternative currency systems.

Applications of this platform over the Internet abound. The value transfer protocol described above needs to be encapsulated in an Internet value transfer protocol that solves the problem of payee spoofing. Such protocols can be constructed, and generally depend on certification of the payee. This is beyond the scope of this paper, as well.

We have already mentioned the idea that in large corporations, internal accounting might be simplified if there were a very easy to administer scrip system. Large corporations are micro-economies wherein commerce is often thwarted by the need to make deals using account transfer techniques involving expensive bookkeeping. We conjecture that a scrip system such as can be constructed using this platform, can be much more efficient both by saving bookkeeping costs, and by promoting more spontaneity of transactions. It can change a planned economy into a more efficient market-based economy.

The final application type that we consider is the area of games. The use of currencies in Internet-based Multi-User-Dimensions (MUDs) and related applications can provide a very significant new strategic dimension. The platform that we describe here can allow this dimension to persist over time and allow the currency to be more portable. These properties can allow serious game playing to include fascinating economic aspects, and can be extended to experiments in commodities trading and stock transactions over the Internet.

8 Notes

We note that for some low-security applications, the use of a smart card is not necessary, or at least the need is less obvious. However, we find it difficult to assess the intensity of the motivation to cheat, and therefore we always presume the use of smart cards or a similar device that provides some non-trivial barrier to cheating.

In some parts of the world, smart cards are becoming very commonplace. In the US, they are arriving slowly. However, the credit card associations seem now to be determined to issue them worldwide over the next few years. There are at least two approaches to smart cards that will promote the use of multiple applications and that might accommodate this platform for privately defined currencies as just one application to co-exist with others such as credit and debit applications and general digital signature apps. These two approaches are Multos from Mondex International and JavaCard from Sun Microsystems. Both of these approaches will illustrate the need to pay strict attention to efficiency concerns, as both systems currently leave little room for individual applications once their respective virtual machines are installed on a smart card. Multos is an operating system for smart cards that features cryptographic services. It is designed to allow multiple applications while addressing the risk that one application might attack or interfere with another. Multos has a virtual machine that interprets a language called MEL. Alternatively, MULTOS can use the JavaCard VM with its API. There are some advantages to this, as JavaCard is a subset of Java, designed to run on smart cards. Unfortunately, many of the security features of Java are absent in the JavaCard specs, and therefore many of the security features of MULTOS or some other operating system are required. MULTOS features aspirations of ITSEC level E6 security. This is the highest security rating in the ITSEC (Information Technology Security Evaluation Criteria) process.

In this paper we have presumed, for simplicity, that there is a single card issuer. In general, we expect that there will be multiple issuers using a common, interoperable infrastructure. This could be accomplished in a number of ways. By forming an association, and offering a service that operates as if there were in fact, one issuer, we can avoid adding complexity to the key management system.

9 Conclusion

The private currency systems that can be constructed using the platform are very easy to use. Consumers need only possess a card to collect currency value, and they can begin to use new currencies without pre-arrangement. Thus, it is very easy for a new customer to begin using a loyalty program. It is also easy for anyone to initiate a currency for almost any application.

The system should be useful for relatively low valued currencies, local loyalty systems, corporate scrip, and games. As we examine and solve various security issues, it may be possible to use a platform like this for major loyalty schemes and even for systems such as stock exchanges.

10 References

1 R.C. Merkle, “A certified Digital Signature”, *Advances in Cryptology*, Crypto ’89 Proceedings, LNCS# 435, G. Brassard (Ed.), Springer, NY, 1990, p. 218.

2 G.J. Simmons, “A survey of Information Authentication”, Chapter 7 of *Contemporary Cryptology*, Edited by G.J. Simmons, IEEE Press, New York, 1992.

3 CEN standard 1546-2, “Identification Card Systems – Inter-sector electronic purse Part 2: Security Architecture ”, European Committee for Standardization, Central Secretariat: rue de Stassart 36, B-1050 Brussels, 1995.

4 D. P. Maher, “Crypto Backup and Key Escrow”, CACM, March 1996.

5 M. K. Reiter, “Secure Agreement Protocols: Reliable and Atomic Group Multicast in Rampart”. In Proceedings of the 2nd ACM Conference on Computer and Communications Security, pages 68-80, November 1994.

6 D. P. Maher, “Fault Induction Attacks, Tamper Resistance, and Hostile Reverse Engineering in Perspective”, Proceedings of the1997 Financial Cryptography Conference, LNCS# 1318, R. Hirschfeld (Ed.), Springer, New York, 1997.