

Periodical Payment Model using Restricted Proxy Certificates

Grigori Goldman

School of Information Technology and Electrical Engineering
University of New South Wales, Australian Defence Force Academy
Canberra 2600, Australian Capital Territory

grigori.goldman@adfa.edu.au

Abstract

In this paper we shall introduce a new electronic payment concept based on the popular direct debit payment model, entitled periodical payments. The direct debit model currently in use online is neither secure nor flexible, and requires a leap of faith by the customer who must trust the merchant to behave honestly. Electronic direct debit request (DDR) forms are not signed by both parties in a binding manner, which means that merchants can change the terms of DDR agreements post-fact. Unsigned DDR agreements give the merchant unprecedented power over customer accounts with little recourse for dispute.

In this paper we shall demonstrate how the use of restricted proxy certificates with cryptographic signatures can be adopted to support a new periodical payment model. A payment policy language is presented that is tailored towards specifying rules that govern precisely how and when merchants can access and transfer funds from customer accounts into their own. Using this model will ensure that mutually signed policies are instantly enforceable on every transaction within a payment period.

There is a fundamental difference between this proposal and other electronic payment schemes. Most such schemes attempt to replicate the features of physical cash such as anonymity, and therefore focus on single payment transactions that simulate cash changing hands. Since direct debit is a popular payment choice, our proposal provides significant improvement to this essentially paper-based payment model that currently does not integrate well in a purely electronic world.

Keywords: e-commerce, payment, periodical, direct debit.

1 Introduction

The notion of electronic payments is not a new one. It dates back as far as early 1980s when David Chaum first presented the concept of using blind digital signatures for implementing untraceable electronic payments. Since then, there has always been much interest in electronic payment systems. Our research has shown that each new proposal focused on two main areas of interest: anonymity of participants of each transaction and the

security of the underlying currency representation (be it digital coins or accounts). For digital coins the primary focus has always been on solving the double-spending problem whereby the same user spends a digital coin twice. For account-based systems, anonymity has always been the more difficult problem to solve.

With the growing popularity of electronic payments a new payment model has emerged. Due to the ever-increasing demand for subscription-based services, direct debit payments have become a very popular method of payment, as is evident from annual reports of both the Reserve Bank of Australia (RBA, 2005) and Australian Payments Clearing Association (APCA, 2005). These types of payments are not confined to subscription-based services and are now available in most industries where traditionally lump-sum payments were required in the past (e.g. insurance industry).

It is no surprise that direct debit payment model found its way onto the Internet and is now being used as an alternative payment method. Fundamentally, however, direct debits are essentially a paper-based payment model. It revolves around a concept of a direct debit request (DDR) form, which is a signed, legally (if not computationally) binding contract between a customer and a merchant.

Personal experience has shown that the direct debit model currently in use online has not changed at all from its paper-based roots. As such, it is neither flexible nor secure and is open to abuse by merchants. Unlike its off-line counterpart, DDR forms are not signed and provide no binding (cryptographic or otherwise) to customers or merchants. This means that in actual fact, merchants may change the terms of DDR agreement post-fact, or in other words, disregard the agreement entirely and use customer account details in whatever way is most profitable. The Reserve Bank of Australia has acknowledged these problems as early as 2001 in its annual report (RBA, 2001), however, the changes that it recommended and introduced are policy driven rather than technical.

Clearly the use of direct debit as an electronic payment method is problematic unless major changes are introduced to restrict merchant abilities to access customer accounts. So far, no other payment scheme to our knowledge has seriously looked into this issue. In fact, most electronic payment research we have examined so far is focused on duplicating the features of physical cash and as such is not appropriate to this model.

In this paper, a fundamentally different approach to electronic payments is taken. Instead of treating each payment transaction as a point-to-point transfer of funds,

simulating physical cash changing hands, payments are viewed as a series of linked transactions. Just like in the direct debit process, customers are required to sign an agreement, which delegates the right to withdraw funds from their accounts to a nominated merchant. The key difference between this proposal and what is currently provided by direct debit is the cryptographic binding of the agreement to the customer and the merchant providing traceability and enforceability.

Another significant advantage of the new approach is its practical use of the payment agreement for enforcing the terms during every single payment transaction. Unlike with the paper-based model where the agreement is only enforceable when legally disputed, or the electronic version where it is not even traceable or binding, the new approach actively assures that each transaction is within the agreed bounds of the policy effectively preventing fraud.

The periodical payment model presented in this paper is driven by the X.509 restricted proxy certificate standard. Restricted proxy certificates are used to delegate permissions to transfer funds from customer accounts to merchants. This certificate becomes the binding artefact that links the customer and the merchant together and that could be used in case of disputes.

Finally, a payment policy language is presented that provides the main instrument for instantaneously enforcing each transaction as discussed previously. This language was designed to describe most common scenarios that are applicable to periodical payments but of course it is flexible enough to accommodate traditional, not periodical payment types as well.

The next section presents a brief overview of the most relevant electronic payment schemes. It also discusses the grid security architecture that successfully used proxy certificates for delegation. Then, follows a detailed introduction of the periodical payment model broken down into two sections: 1) the initial delegation of the payment credential to the merchant and 2) the use of that credential to initiate a single payment transaction. The semantics of the payment policy language will be discussed next. Finally, a brief discussion of anticipated future work is presented.

2 Related Work

There are numerous electronic payment systems that have been proposed and developed over the years. However, none approach the problem from the standpoint of periodical payments. As such there is very little work that has directly influenced the direction of this paper.

Most of the work that deals with notational, account-based electronic payments has focused on three major areas of research: authentication of participants, security of transactions and in some cases anonymity. No explicit attempts to solve the problem of delegation in electronic payment environments have ever been seriously considered.

There is some research that stands out as relevant in their approach to payments in general and could be considered

as the first building blocks for the implementation of periodical payments. For example, (Bellare et al., 1995) and (Bellare et al., 2000) presented an interesting protocol entitled i-Key-Protocol (iKP), which discussed the possibility of integrating a new secure approach to payments into an existing payment clearing infrastructure. Just like the proposal discussed in this paper, iKP uses the payment gateway concept as the mechanism for interacting with merchants. Its purpose is to convert payment artefacts into format that will be recognised by the legacy systems.

Similarly, iKP also relies heavily on certificates as the authentication and authorisation mechanism. However, the authors do realise that the level of PKI acceptance and market penetration is not sufficient to allow effective adoption of the mechanism. Hence this protocol introduces a staged integration approach whereby the first stage requires no PKI at all while the later stages gradually introduce certificates.

Another interesting notational scheme is the anonymous Internet mercantile protocol presented by (Low et al., 1994). This protocol does not address the unique requirements for periodical payments directly, however, it is one of the very few schemes that deliver a form of multi-transactional support. It works by allowing customers to set up session (i.e. temporary) accounts with merchants, which can be debited by the merchants without explicit involvement of the customers. Whenever the funds in such accounts run out, the customer is responsible for injecting more money if the relationship with the merchant is to continue.

The main motivation for designing the mercantile protocol was not to enable periodical payments. Instead it arose from the desire to streamline the payment process so that it would be quicker and more convenient for the customer and merchant.

There are significant conceptual differences, however, between the protocol presented by (Low et al., 1994), and the one discussed in this paper. One major limitation of the session account concept is the fact that all of the funds have to be committed before the transaction takes place. This, in a fact, contradicts our current definition of periodical payments, which allows customers to split a single large payment into multiple smaller ones, distributed across multiple payment periods. This is important since in some cases the motivation for choosing periodical payment option is because of the lack of funds to complete the entire transaction in one hit.

Since the funds are committed at the beginning of the transaction, this protocol therefore fails to deliver on another important requirement for periodical payments. The customer is required to relinquish control of the funds to the merchant and loses all control of how those funds are managed. This means that given a malicious merchant, the funds can be taken without permission. No effort to restrict access to funds is made with the only advisable safeguard being to transfer only small amounts of money into such accounts.

Due to the lack of appropriate solutions to the periodical payment problem, the work on grid security was closely

studied so that some of the underlying concepts could be adopted for use in electronic payments. For example, (Welch et al., 2004) presented a discussion on the use of X.509 proxy certificates for enabling delegation in grid environments. This work followed the general discussion on grid security services presented by (Welch et al., 2003).

Significant amount of research and development towards improving and standardising the proxy certificate specification (Tuecke et al., 2004) was accomplished through the development of the grid security services. Its major contribution, the Open Grid Services Architecture (OGSA), and in particular, its Grid Security Infrastructure (GSI), has delivered a set of standard tools that use proxy certificates for authentication within a complex, heterogeneous environment.

The grid architecture presents a very compelling reason for using proxy certificates since it requires its users to authenticate themselves to various services distributed across a wide area network. Furthermore, its sole purpose is to allow users to execute computationally expensive operations, which can potentially take hours, or days and that require minimal supervision from those users. When presented by (Koufil and Basney, 2005), these tasks were commonly referred to as batch jobs because they are not executed when submitted but are scheduled and executed when appropriate hosts are found and enough resource have been allocated to execute them.

It is clear how the use of proxy certificates within grid environments is applicable in periodical payments. Periodical payment requirements on delegation and authorisation can be seen as a subset of the overall problem that the grid security infrastructure is trying to solve. The general principal behind the use of restricted proxy certificates is to delegate both the issuer's identity and some subset of issuer's privileges to the bearer. For periodical payments, the resources for which access is granted is always the customer's bank account, while the privileges being delegated always refer to the amount of funds that can be transferred and who can do so.

Some common issues that are relevant in grids are not relevant in periodical payments, which simplify the paradigm. For example, dynamic creation of new entities within the network, which is an important concern in grids, is not an issue since it is always known before each transaction, which parties are involved and once the payment contract is negotiated no amendments are allowed.

In the next section, a complete model of the periodical payment process is presented. This scheme uses the restricted proxy certificates developed as part of the grid security project but instead of authentication their role is to convey customer account access authorisation information.

3 Periodical Payment Process Using Restricted Proxy Certificates

The complete periodical payment process is somewhat involved. Unlike the traditional model, which assumes

immediate transfer of funds, periodical payments do not. Instead, this process works in a schedule-type framework, whereby the initial (and only) customer-merchant interaction only establishes the terms of each scheduled transaction. Before discussing the details of the periodical payment protocol it is important to understand this fundamental idea.

The periodical payment policy is the core of the model. This policy is essentially an agreement, a contract between a customer and a merchant. The merchant promises to provide a customer with a service while the customer agrees to pay for that service on regular basis. This policy is essential to this process because it places restrictions on the delegated credentials that the customer must give the merchant so that all subsequent transactions can be performed without further customer intervention.

The important qualities for the policy language that are of particular interest are its simplicity and flexibility. It is envisaged that the contents of the payment policy would need to be presented to customers in a human-readable form so that manual validation could be performed. As such, language simplicity is of the highest importance to ensure that customers can easily visually inspect each periodical payment contract and understand it.

While simplicity is of particular importance, language flexibility is also crucial. The policy language must contain sufficient expressiveness to describe most common periodical payment scenarios. For example, when delegating credentials, it is most likely that customers would be interested in the following restrictions:

1. The merchant will only charge customer account for services provided (or to be provided) to the customer.
2. The merchant will only charge the agreed amount or within a specified range (e.g. anything under \$200, etc.)
3. The merchant will only charge customer account once per agreed payment period.
4. The merchant will cease to charge customer account upon completion of the contract or as soon as the service for which it was established is no longer provided.
5. The merchant will cease to charge customer account on request.

It is clear how the above assertions could be used by a payment gateway to validate each transaction initiated by the merchant using customer-delegated credentials. Later in this section, it will be demonstrated how each of the above assertions can be easily expressed using our experimental policy assertion language. For now, let's examine the two stages of the periodical payment process: 1) the delegation of the payment credential (including signing of the policy document), and 2) initiation of a payment transaction (i.e. transfer of funds from customer to merchant account).

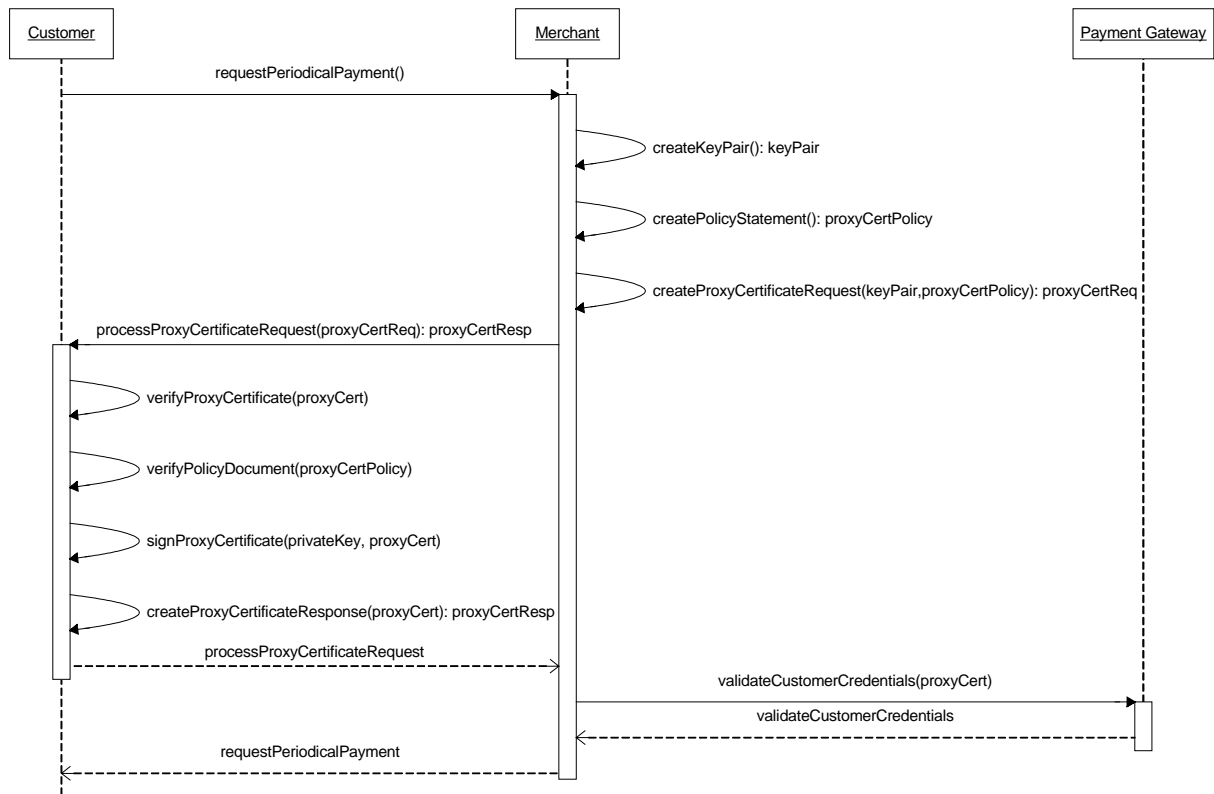


Figure 1: Payment Certificate Delegation Process

3.1 Delegation of Periodical Payment (Proxy) Certificate

The delegation of a periodical payment certificate is the first step in the payment process and is the only one that involves the customer. It is based on the X.509 proxy certificate delegation process defined in (Tuecke et al., 2004). The only addition to the standard protocol that is necessary is the creation and acceptance of the policy document that forms the foundation of the payment certificate. The delegation steps are depicted in Figure 1 and can be described as follows:

1. Once the customer requests a periodical payment option, the merchant generates a new public-private key pair.
2. The merchant then creates a policy statement.
3. Using the key pair created in step 1, the merchant creates a restricted proxy certificate request. This request must conform to the specification described in (Tuecke et al., 2004). It will also contain the policy created in step 2.
4. The request is sent to the customer for consideration (i.e. the customer must process the request to examine the policy and to sign the certificate).
5. The customer must verify that the proxy certificate request is valid, i.e. it is not an end-entity certificate and all of its mandatory fields have been correctly set. Normally, the policy document is an optional field, however, for our purposes it is mandatory.
6. The customer must review and verify the terms of the policy document received from the merchant. It can either try to renegotiate the terms of the policy, reject it or proceed with the next step.
7. The customer must sign the proxy certificate with its own private key, create a response message and send it back to the merchant.
8. Finally, the customer can create the proxy certificate response message and forward it to the merchant.
9. Once the merchant receives the response, it needs to validate the authenticity of its signature. This task is delegated to the payment gateway that is capable of validating all customer credentials. If a positive response is received, the merchant can use the proxy certificate subject to its policy.

The final step (step 9) of this process concludes with the merchant obtaining possession of the restricted proxy certificate which, when presented to a payment gateway, will enable it to transfer funds from customer account to its own. The policy contained within this certificate is the artefact that is used by the payment gateway to make its decision whether to allow the transfer to proceed or to reject it.

There is a significant complication with the above process. Upon detailed analysis it was observed that a certificate with a restriction policy is not sufficient for the payment gateway to determine whether a single transaction is within the bounds of the policy agreement. That is, since a policy can dictate a long period between each transaction (e.g. a month or a quarter), a policy alone does not convey the necessary information to

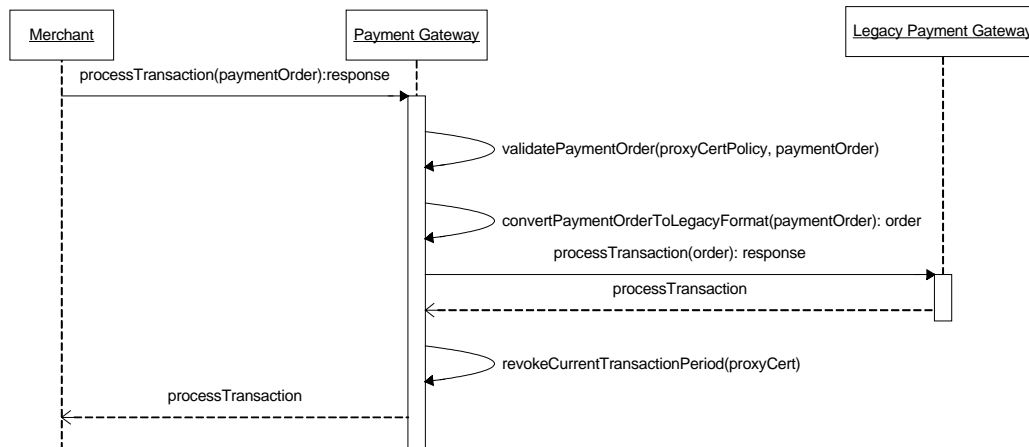


Figure 2: Single Payment Transaction Process

uniquely identify a transaction. This is a classic double-spending problem. In this case, the payment gateway cannot use a certificate policy alone to determine whether a particular transaction was already performed within a specific payment period.

3.1.1 Solving the Double-Spending Problem

Clearly the double-spending problem as discussed so far is precisely the reason why a new periodical payment approach is being presented. It is one of the most important issues that cannot be sufficiently addressed by the current direct debit model.

Using proxy certificates for payment authorisation, however, provides the periodical payment model with various convenient ways of solving this problem. The simplest solution to the problem is for the payment gateway to record each transaction performed by the merchant. This log of transactions in combination with the certificate policy is sufficient to determine whether an individual transaction is valid. The payment gateway must simply ensure that all assertions within the policy are met and that the merchant has not performed a transaction using this certificate in this payment period by consulting the transaction logs.

This solution demands a great deal of the payment gateway. It is inefficient to store all transactions that have been performed and after closer analysis it was observed that it is not necessary either. The payment policy, which will be presented in depth later in this paper, declares one important element called *period*, which can help a great deal in reducing the amount of data that a payment gateway must store.

The *period* element declares the interval between each transaction (e.g. week, month, bi-monthly, etc.). This is precisely the information that the payment gateway needs when logging transactions.

The design that was adopted for the logging mechanism is based on the concept of revocation similar to certificate revocation lists described in (Housley et al., 2002). The full credential cannot be revoked since no more transactions would be possible and the periodical

payment model will be broken. Instead, the payment gateway should only be interested in the last transaction that was performed by the merchant using a payment credential. Since the credential cannot be revoked, the payment gateway can use the *period* value declared in the policy to revoke the use of that credential for that period. When used a second time within the same period, the payment gateway can check its revocation lists and identify double-spending attempt.

The format of the revocation lists for payment certificate periods is simple. It needs to contain two values. The first value must be the X.509 certificate unique identifier and the second the period expiry date. The period expiry date is the last date when the policy can be used to access customer funds within a particular period. The only complexity is the determination of the expiry date based on the policy period attribute. However, this is not a major issue since the payment gateway can reuse its policy validation process, which must perform a similar function, to extract this date.

3.2 Payment using Periodical Payment (Proxy) Certificate

The actual process of initiating a payment transaction is simple. The payment certificate, which is just a normal X.509 proxy certificate can be passed to the payment gateway via Transfer Layer Protocol (TLS) establishing a secure connection to the gateway server. No modification to this protocol is needed. It will handle all of the necessary proxy certificate authentication-authorisation using the standard path validation process described in (Housley et al., 2002). Post authentication process is depicted in Figure 2 and can be described as follows:

1. Once an encrypted channel is initialised, the merchant can send the payment order for processing of the transaction. This can be a simple XML object that contains payment details such as from and to accounts and amount.
2. Having received this information, the payment gateway must validate it (i.e. validate payment order)

against the policy document it received via a payment certificate. It must verify that:

- The certificate has not expired.
- This transaction is within an acceptable period as declared in the policy.
- The current payment period for this certificate has not been revoked, that is the merchant is performing this transaction for the first time within this payment period.
- The payment order amount is within the acceptable range as declared in the policy.

Any errors in validation will force the payment gateway to reject the payment order.

3. Upon successful validation of the payment order against the policy, the payment gateway can proceed with its processing. At this time, it is assumed that an existing payment clearing infrastructure is most likely to be used for actual funds transfer. As such, the role of the payment gateway in this scenario is to convert the payment order into a format that the legacy clearing infrastructure can understand.
4. Converted payment order is then submitted to the legacy system. Clearly in this case any problems that occur during payment processing once submitted are out of scope.
5. The final step that the payment gateway must perform is to revoke the current payment transaction period for the certificate. Clearly this step cannot fail, otherwise this will give a merchant an opportunity to submit a duplicate payment order and charge its customers twice. For this reason, great effort is required to assure that payment submission and the revocation process are atomic operations.

Upon completion of this process the payment gateway may respond to the merchant by issuing a response (i.e. receipt) message that it can use as a reference.

4 Periodical Payment (Proxy) Certificate Policy Language

Periodical payment policy language is expressed as an XML document and it is surprising simple. It requires only three custom data types with just four different XML elements. Each element represents an assertion that the payment gateway must evaluate to true before processing a transaction.

Assertions By Type	Element
Date	<not-before-date> <not-after-date>
Periodical Transaction	<transaction>
Payment	<payment>

Table 1: Policy Elements

The first and most straightforward assertion type is the date type. Naturally, its purpose is to provide a mechanism for specifying concrete dates that can be used

to declare contract boundaries where appropriate. Within the policy XML this type is represented by two elements: <not-before-date> and <not-after-date>. Each element contains a single attribute, *value*, whose contents must conform to the XML date format defined by the (XMLSchema, 2005) specification.

Attribute	Required?	Value
Value	Yes	xsd:date

Table 2: Date Attributes

For example:

```
<not-before-date value="2007-01-01"/> and
<not-after-date value="2007-12-31"/>
```

The date assertions have an important role within the policy. Not only can they be used as global assertions that specify the contract boundaries, they can also be used as constraints placed on other assertions within the same policy. Constraints will be examined in more detail later in this section.

In addition, to the date assertion type, the policy declares a payment type. This type is expressed by a single XML element <payment>. Its purpose is to define the payment options that are available to the merchant. That is, using this assertion the payment gateway can determine whether the amount of funds that the merchant wishes to transfer from the customers' account is within the agreed range.

The payment assertion is capable of representing various amounts and currencies. It can be used to declare a single, non-changing amount or it can specify a range of acceptable amounts. To achieve this, it uses three attributes: *currency*, *amount* and *type*.

Attribute	Required?	Value
Amount	No	xsd:decimal
Currency	Yes	AUD USD ...
Type	Yes	Fixed Limit No-limit

Table 3: Payment Attributes

The purpose of the currency attribute is self-explanatory. Since it is possible for transactions to be international, a mechanism is required to specify which currency the merchant is dealing with.

The amount attribute is closely related to the value of the type attribute. Depending on whether the type is *fixed* or *limit*, the meaning of the amount attribute changes. For a fixed type, the amount value is static. That is, the merchant can only withdraw the exact amount declared in the policy. Anything else must cause a validation error. For example:

```
<payment currency="aud" amount="20" type="fixed"/>
```

For limit type, on the other hand, the amount value indicates the upper boundary of the acceptable range of values. In this case, anything up to and including the value is considered acceptable. For example:

```
<payment currency="aud" amount="100" type="limit"/>
```

Finally, the last and undoubtedly the most complicated assertion type is the periodical transaction type. Within the policy XML this type is represented by a single element <transaction>. Its purpose is to declare the interval (i.e. period) between each allowed transaction. It must do so within a single expression, which makes it slightly more complicated than the previous two assertions.

It is clear why the transaction assertion is complex. Within a single rule it must be able to represent the complex behaviour that is inherent in periodical payments. It must be flexible enough to describe all transactions within the scope of the contract. As such, this assertion requires four attributes: *period*, *day-of-week*, *week-of-month*, and *day-of-month*.

Attribute	Required?	Value
Period	Yes	Daily, Weekly, Fortnightly, Monthly, Quarterly, Half-yearly, Yearly
Day-of-week	No	[Mon, Tue, ... Sun]
Day-of-month	No	[1...31]
Week-of-month	No	[1...5]

Table 4: Transaction Attributes

The period attribute has already been briefly introduced earlier in this paper. Its purpose is to indicate the agreed frequency of transactions, for example weekly, monthly, quarterly, or yearly. By itself, however, it is not sufficient. More information is required by the payment gateway to determine when precisely a transaction can be processed.

Day-of-week, week-of-month and day-of-month are the attributes that can be used to provide the payment gateway with that extra information that it needs. Their use is determined by the value of the period attribute. For example, for a weekly period, only the day-of-week attribute is required. Its value must be a valid weekday three-letter acronym. For example:

```
<transaction period="weekly" day-of-week="mon"/>
```

For a monthly period, on the other hand, day-of-month attribute can be used to specify on what day of every month a transaction can be processed. For example:

```
<transaction period="monthly" day-of-month="1"/>
```

As you can see there are various possible combinations, each one requiring the use of different attributes.

4.1 Coping with Odd Transactions

The fundamental principle underlying the payment and transaction assertions is that there will only be a need for one assertion each to describe all of the necessary

conditions for a contract. This assumption is true in most cases; however, there could be instances when a policy needs to declare an odd transaction to cope with a special case. For example, for a fixed term contract, there could be a need to declare an odd transaction to handle the repayment of the remaining amount upon termination of the contract (i.e. as part of the last transaction).

To handle odd transactions additional payment or transaction assertions need to be declared within a single policy. Having two or more assertions of the same type within one policy, however, is problematic because it introduces ambiguity as to which one the payment gateway should validate against for a particular payment period. To remove this ambiguity, the concept of constraints was introduced.

A constraint is not a stand-alone assertion. Instead, it is an optional sub-element of the payment or transaction assertions. Within the policy language it is expressed as an XML element <constraints>. Within each constraints element at least one date assertion must be declared, either <not-before-date> or <not-after-date>, or both. When validating a payment order against a policy the payment gateway can distinguish between assertions of the same type by checking whose constraints make it applicable within the current payment period.

The following example depicts two payment assertion declarations whose constraints define which payment period they apply to:

```
<payment currency="aud" amount="80" type="fixed">
  <constraints>
    <not-after-date value="2007-11-30"/>
  </constraints>
</payment>
<payment currency="aud" amount="50" type="fixed">
  <constraints>
    <not-before-date value="2007-12-01"/>
  </constraints>
</payment>
```

By introducing the concept of constraints additional policy validation logic is needed. This checking is important because even with constraints it is still possible to declare two assertions of the same type that overlap within a payment period. Policy validation must identify this scenario and reject any contract that does not uniquely specify a single assertion per payment period.

5 Periodical Payment Model Implementation

In this section we shall briefly describe periodical payment components that have either already been implemented or are due to be implemented in the near future.

5.1 Periodical Payment Policy Parser and Validation

To date, the first draft of the policy XML schema has been completed. Its corresponding parser and validation logic have been implemented using Java and Apache XML-Beans toolkit. Using XML-Beans allows us to bind

XML types to Java objects. XML-Beans is used to compile the policy XML schema that generates both the XML parser and the corresponding Java data types.

When parsing policy XML contracts the parser performs semantic/syntactic validation of the XML. However, this validation is insufficient. Additional validation logic was implemented to check policies for ambiguity. A custom validator, written in Java, is used for checking that all elements that have constraints are not in conflict with each other. In addition, element attributes that are inter-dependant are checked to ensure that their values are meaningful in the context in which they are used.

5.2 Customer and Merchant Libraries

Periodical payment process is composed of two distinct communication components. The first component dictates the communication between the customer and the merchant while the second one controls merchant-payment gateway interaction. Considering that our primary focus to date was the XML policy schema specification and its validation, at the time of writing the communication components have not been implemented.

For the customer-merchant interaction client-side libraries are needed that can securely communicate with the merchant payment systems. Unfortunately, currently there are no standard mechanisms for implementing such client libraries. We are considering implementation of a browser plug-in that will detect periodical payment transactions and will facilitate customer interaction with merchant payment systems.

Implementation of the merchant to payment gateway interaction component will be completed in the near future using web services. Web services technology offers two important advantages. Firstly, this technology integrates well with the secure socket layer (SSL) protocol. We heavily rely on it for establishing a secure channel between a merchant and a payment gateway. Also, it is the mechanism that allows the merchant to send the policy statement to the payment gateway via a proxy certificate.

In addition, web services are a proven technology. They have been used in numerous commercial applications where decoupling of components and interface flexibility is of particular importance. Unlike existing payment gateway interfaces, using web services allows us to generalise this interface, which means that merchants will no longer require proprietary libraries supplied by the payment gateway providers to process transactions.

Existence of readily available web services implementations gives us confidence that this technology can be quickly adopted for periodical payments. Its performance, however, is a crucial factor that will impact its acceptance as an electronic payment solution. Therefore, it is our intention to develop a prototype implementation of the payment gateway web services interface, and to conduct initial performance testing of it to validate that it can process a suitably large number of concurrent transactions.

6 Future Work

As was mentioned in the previous section, the next task on our agenda is the development of the payment gateway web services interface and corresponding merchant libraries. Using this implementation we shall analyse the performance of the payment model. In addition, the necessary customer side libraries will also be developed to allow for complete, end-to-end test coverage of the payment process.

The periodical payment model is a complex scheme that requires infrastructure-wide changes to be effective. Presently, it is unreasonable to expect all consumers to be able to properly protect their private keys. The adoption of smart cards is slow and most current computers are not equipped to read them. As such, it is unlikely that any technology that makes demands on client side PKI will work. Therefore, as an alternative, it will be examined how conventional username/password technics can be adapted to the periodical payment model presented here.

Another potential area that needs closer scrutiny is converting the periodical payment policy into human readable form. There are two important reasons for this. Firstly, as it was stated previously, we consider the customer's ability to visually inspect the policy contract an important part of the policy validation process. Unless this information is presented to a customer in an understandable way, the customer's signature will have no meaning.

Another improvement that is closely related to the one above, is exploring mechanisms for customers to negotiate the terms of the policy contracts. Once each policy is presented to the user in a readable form, a mechanism is needed that allows the customer to make changes to the policy and submit them to the merchant for consideration. Merchants need ability to declaratively specify the boundaries of acceptable amendments so that change requests can be automatically processed and accepted, rejected or alternative counter-amendment proposed. A communication protocol needs to be established that allows both parties to participate in this negotiation in a secure manner.

Also, an interesting area that needs further improvements is the payment gateway transaction validation process. The current model requires the payment gateway to keep a transaction revocation list to prevent double spending, which is an expensive task. Ideally, the payment gateway should be stateless and require no local storage of transaction data. This means that further investigation needs to be performed in determining whether more information can be encoded in the payment credentials that will reduce the need for local data to be maintained by the payment gateway.

In addition, it should not be overlooked that regardless of its intended purpose, the periodical payment model is just another payment scheme. As such, some investigation is warranted into its application as a conventional, non-periodical payment scheme. Of course depending on the overall complexity of using this particular approach as a normal payment mechanism it may prove impractical.

Finally, even though periodical payments do not fit well the traditional electronic payments approach it is still quite interesting to examine potential addition of anonymity to the scheme. In most electronic payments research a great deal of importance is placed on anonymity and as such should not be overlooked for periodical payments although once again their practical use might be of limited interest and acceptance.

7 Conclusion

Periodical payments are radically different to traditional e-commerce payments. Payment transactions require no customer involvement during each transaction and empower the merchant to initiate payment transactions at whichever time they see convenient. Transfer of control of customer personal banking details to merchants creates an overwhelming need for a better security mechanism than what is currently available.

Periodical payment model represents an important change in direction for electronic commerce payments. They fill the security and useability gap that the direct debit model used now is incapable of doing due to its paper driven design. Neither can the existing solutions offer value due to their fundamental focus that place anonymity and non-relinquishing of control as priority.

The periodical payment model presented in this paper builds on a strong cryptographic foundation. It uses the existing and proven X.509 standard for securing payment channels and for providing customer, merchant and payment gateway authentication. The use of the restricted proxy certificate concept allows us to implement transfer of control mechanism that enables the customer to securely transfer account access rights to the merchant.

The use of restricted proxy certificates serves another purpose. It can be used to carry the periodical payment policy. This policy replaces the traditional direct debit request form as the customer-merchant payment agreement that the customer must sign. Our ability to easily verify the legitimacy of the payment certificate and to determine the applicability of policy assertions to the current transaction is what gives this model its great flexibility and allows safe transfer of control through delegation.

8 References

APCA (2005): Annual Report. Sydney, Australian Payments Clearing Association.

Bellare, M., Garay, J., Hauser, R., Herzberg, A., Krawczyk, H., Steiner, M., Tsudik, G. & Waidner, M. (1995): iKP - A Family of Secure Electronic Payment Protocols. *Proceedings of the First USENIX Workshop on Electronic Commerce*. New York, USA, USENIX.

Bellare, M., Garay, J. A., Hauser, R., Herzberg, A., Krawczyk, H., Steiner, M., Tsudik, G., Van Herreweghen, E. & Waidner, M. (2000): Design, implementation, and deployment of the iKP secure electronic payment system. *IEEE Journal on Selected Areas in Communications*, 18, 611-627.

Housley, R., Polk, W., Ford W. & Solo D. (2002): Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. *RFC3280*. Internet Engineering Task Force (IETF).

Koufil, D. & Basney, J. (2005): A credential renewal service for long-running jobs. *Proceedings of the 6th IEEE/ACM International Workshop on Grid Computing*. Seattle, Washington, USA, IEEE Computer Society Press.

Low, S. H., Kristol, D. M. & Maxemchuk, N. F. (1994): Anonymous Internet Mercantile Protocol. *Technical Report*. Murray Hill, New Jersey, AT&T Bell Laboratories.

RBA (2001): Payments System Board. *Annual Report*. Sydney, Reserve Bank of Australia.

RBA (2005): Bill payments and Automated Teller Machines. *Annual Report*. Sydney, Reserve Bank of Australia.

Tuecke S., Welch, V., Engert, D., Pearlman, L. & Thompson, M. (2004): Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile. *RFC3820*. Internet Engineering Task Force (IETF).

Welch, V., Foster, I., Kesselman, C., Mulmo, O., Pearlman, L., Tuecke, S., Gawor, J., Meder, S. & Siebenlist, F. (2004): X.509 Proxy Certificates for Dynamic Delegation. *Proceedings of the 3rd Annual PKI R&D Workshop*. Gaithersburg MD, USA, NIST Technical Publications.

Welch, V., Siebenlist, F., Foster, I., Bresnahan, J., Czajkowski, K., Gawor, J., Kesselman, C., Meder, S., Pearlman, L. & Tuecke, S. (2003): Security for Grid services. *Proceedings of the 12th IEEE International Symposium on High Performance Distributed Computing*. Seattle, Washington, USA, IEEE Computer Society Press.

XMLSchema (2005): XML Schema Specification, World Wide Web Consortium (W3C), <http://www.w3.org/2001/XMLSchema>. Accessed 3 Aug 2006.