Confidentiality and Anonymity Analysis of On-line Payment Protocols

Salvador Mandujano, Clay Shields
sam@cerias.purdue.edu, clay@cerias.purdue.edu
Department of Computer Science
and CERIAS
Purdue University
West Lafayette, IN. 47907 USA

September 29, 2000

Abstract

The evolution of e-commerce has generated multiple payment systems that customers and businesses use to perform commercial transactions over the Internet. These systems were designed to provide their users with secure transactions that guarantee the safe transfer of funds, but most of them fail to protect the privacy and anonymity of the parties involved. This paper presents an analysis of the privacy features of some of the most representative electronic payment protocols, namely, credit card-based and digital cash systems. It also describes novel security technologies that can be used to enhance lax controls on these systems and discusses the advantages and disadvantages of using electronic cash, credit card systems and electronic wallets as alternative payment methods on the web.

1 Introduction

Web technologies today allow websites and their advertising affiliates to record private information from customers visiting their websites. These systems usually store data like browser type, connection IP address and previously accessed web pages. Whenever a web transaction is completed, these sites therefore get to know part of the identity of the users. If the user provides personal information, as may be required for a transaction, then it is possible to link this personal information with the user's network identity [3] . Even when the promise from these vendors is not to sell, share or rent this information to anyone, the fact is that repositories of confidential data are being created for purposes that do necessarily relate directly to the business the site conducts. This, to some extent, violates the privacy of the users as combining different pieces of information could uniquely identify machines and individuals.

This paper focuses on two particular points of electronic protocols: *confidentiality* and *anonymity*. We describe the fundamentals of some payment systems from the point of view of these two features. While several schemes have been created to guarantee that a payment being sent by a user to a merchant over the Internet is safely transmitted without exposing it to attackers, each of the existing systems has strong

and weak points in terms of maintaining privacy. We also indicate the need to have an acceptable degree of anonymity for certain types of purchases so that people can get products without having to reveal their identity.

The reminder of this paper is organized as follows. Section 2 establishes some assumptions and ideas useful to the discussion on anonymity and confidentiality. Section 3 describes some representative payment protocols and talks about their security features. Section 4 presents security controls and protocols that can help strengthen electronic payment systems. Our concluding remarks are in Section 5.

2 Definitions and considerations

For the purpose of our discussion, we now define privacy, confidentiality, and anonymity from the perspective of payment systems. Alternative definitions and further discussion on these concepts can be found in [3, 4, 5, 6, 7, 9].

Confidentiality. For the purpose of electronic payment systems, confidentiality means protecting the payment information from being read or copied by anyone who is not an authorized participant in the transaction. This is the standard goal of most electronic payment systems.

Privacy. Privacy is the ability of an individual to maintain control of their personal information, so that the individual chooses when and to whom that information can go. This information might be things such as the individual's name, address, financial information, or simply consumer preferences.

Anonymity. Anonymity can be defined as the quality of being part of some large group, and being indistinguishable from other members of that group. Anonymity allows for privacy by permitting a user to release some personal information without the receiver being able to link that information with other information about that user.

As we will discuss later, banking institutions and financial third parties providing electronic payment services are likely to be vulnerable to *privacy attacks* due to the amount of confidential data they need to store. In this kind of attacks, outsiders try to break into the data banks of these institutions so that they can get private customer information that help them impersonate a user, use their financial resources or, simply, target a good objective for extortion or electronic fraud. Protocols like *Digicash* and *Ecount* deal with this problem by not having a general customer database. They just make digital coins available to purchasers using anonymous methods so that a buyer never has to identify herself to the merchant nor to the financial institution when she gets her e-coins by cash. It is important to mention that totally-anonymous e-commerce is only possible for things that can be supplied anonymously since buying goods that need to be shipped to an address would eventually reveal part of the identity of the buyer. For financial purposes, anonymous e-commerce is more suitable for small transactions. Being able to move around large amounts of money anonymously over the Internet can lead to money laundering and fraud when skipping national or international taxation laws. However, customers still do have the right to get products anonymously without having to give out their name and address. These digital cash systems do provide anonymity to the purchasers but when a payment is sent, an eavesdropper could intercept it and money could literally get lost if no recovery system exists. The use of encryption on electronic money prevents it from being spent by the attacker but the customer has no way to recover her coins once they are sent to the merchant. The following analysis will help put all this into perspective highlighting the importance of having trustable payment systems.

3 Electronic payment systems

We discuss now the structure of some electronic payment systems. *Cybercash, Digicash, Magic Money, EMoneyMail, First Virtual, PayPal, PaybyCheck, Ecount, SET*, and *OpenMarket* were studied to find the security features common to them making a clear distinction between e-cash and credit-card based systems. The details of these protocols are taken from [8] and from their respective business websites. Even when the specifics of some very recent protocols are not yet available [27, 28], they do share an overall structure with well-known protocols. Here we present the most important ones in terms of design representativity according to the two security features we are interested in.

First Virtual [8, 10, 12]. *First Virtual* (FV) is based on the use of credit cards and e-mail infrastructure (see also *PayPal*, [28]). Since one of the main problems in electronic payments is sending credit card numbers over unsecure channels, the FV protocol substitutes those numbers by affiliation personal identification numbers called *virtual PIN's*, or *VPIN's*. In the event a VPIN is intercepted by an outsider, it cannot be used to directly make charges to the owner's account. This is because FV is the only entity that stores the VPIN's to credit card number mapping, so its databases need to be accessed first in order to get a credit card number. FV concentrates all this information as the central authority it is. Any merchant or user who wants to make use of the system needs to create an account with FV first. FV also works as an arbiter for all the transactions between merchants and buyers. Following is a description of the protocol (see Figure 1).

1. The customer opens an account with FV. She provides her name, address, phone number, e-mail address and payment information (i.e., whether she will pay by credit card, check or debit card).

2. The merchant does the same by opening a FV account.

3. The customer selects a product from the merchant's website or catalogue and sends an e-mail message requesting the purchase. This message includes her VPIN and the product's code.

4. Upon receipt of the message, the merchant checks with FV whether that VPIN corresponds to an actual account. This check can be performed over the phone or using a website (see [10] for details).

5. Once confirmed, the merchant sends e-mail to FV including its own VPIN, the customer's VPIN, product code, product description, and the price.

6. Before making any charge to the customer's account, FV sends a confirmation message to the customer. This indicates the product being purchased, the merchant's name and the price, and it asks the customer whether she accepts the charge.

7. The customer has three possibilities to answer: Accepting the charge, denying the charge in case something on the order is incorrect, or denying the charge as indicative of possible fraud.

8. FV then receives the answer (assuming everything was correct) and sends a copy of the customer's answer to the merchant.

FV is simple and functional but it does not protect the identity of the parties involved in a transaction. Since it uses common mail client and servers, the e-mail addresses of customer and merchant are always visible to eavesdroppers. It is successful, though, in hiding credit-cards numbers by using VPIN's. This prevents the risk of having an eavesdropper getting credit card information over the communication channel.

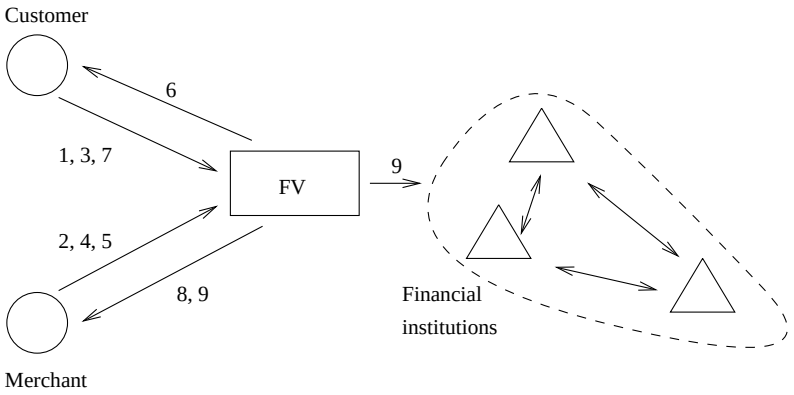

Figure 1: The First Virtual payment protocol

The central data repository VPIN's would have to be compromised in order to get the credit-card number bound to a VPIN code. It is important to notice that the main design flaw FV has is that it does not use any sort of encryption. In step 5, several pieces of private information are sent out to FV including VPIN codes and product information. Other payment systems, like SET and Open Market, do not make this information available to observers and therefore are more secure from the privacy standpoint [19, 20].

Steps 3, 5, 7 and 8 reveal the electronic addresses of merchant and buyer. If an attacker is capturing these messages, she can use them as a starting point to determine the individual identity of the person carrying out the purchase. If these e-mail accounts happen to be from a public mail system this attack would succeed more easily since most of these systems provide search engines to get the address or name of the users. Impersonation is possible if the VPIN of a user and her address are revealed, and if the attacker spoofs her e-mail address and intercepts messages from FV preventing them to reach their intended receiver, or if the receiver is not reading her mail. Assuming that no repudiation feature is available in the mail system, the user whose VPIN was compromised could be charged with the purchases made by the attacker.

FV could obtain great benefits from multiple security controls (see section 4 below). Any suitable encryption system, such as SSL, would provide customers and vendors with the degree of confidentiality they are missing. The price of this would depend on the type of cryptographic protocol chosen, but it has always a cost in performance that greatly varies if public or a private-key scheme is implemented. A repudiation feature build on the mail server that allowed users to deny confirmation messages would help strengthen the transaction risk protection users have. The problem of using e-mail is that when the password authentication system of the mail system is defeated, an attacker can send mail pretending being the owner of that e-mail address. A second 'purchase password' or 'purchase code' stored on FV servers would help create a repudiation feature in which a user needed to send purchase messages from a specific address always including her purchase code. Impersonation would be more difficult now but the user would have to handle two passwords when using FV. Using encryption is also a solution to non-repudiation. A customer could sign every e-mail message requesting a product through FV so that the merchant could verify the request before proceeding with the operation. An attacker would be unable to forge a request if she does not have the private key to sign the purchase messages.

FV could also benefit from *re-mailers* [7, 8, 22] used along with encryption. The purpose of re-mailers is to forward messages on behalf of a user so that the recipients cannot tell their origin. Some re-mailers that

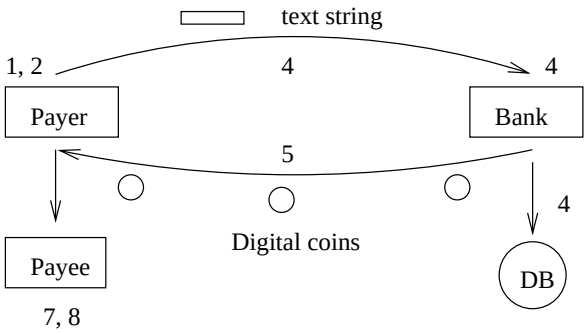

Figure 2: Digicash

could be used with FV do provide replying capabilities by pseudonym. These systems are less secure than total-anonymous re-mailers (i.e., the ones that do not use alias or pseudonyms), but can be well used for this purpose. Delivery latency would be affected, but better security would be achieved. Anonymizing systems make it much more difficult to determine the identity of a user and do provide protection against outsiders as well. It is important to notice that arbitrated protocols like FV have always bottlenecks (namely, the arbiters) so adding these controls to an e-mail-based system will have an impact on the traffic going through these bottlenecks, as they will have more processing work to do.

Digicash [18]. From a mathematical standpoint, '*Digicash* is the only protocol that can be shown to allow a customer to remain anonymous once the payment is complete' [8]. Another electronic cash system called *Magic Money* [27] also provides this quality by using Chaum's *blind signing*, but we discuss Digicash as a representative model. Digicash does not require, as credit card systems do, that the identity of the customer be made available to the vendor or payment service provider. It is an untraceable on-line virtual-cash system that uses digital signatures - *public key cryptography* - to provide authentication, non-repudiation, data integrity, and confidentiality at a very high level. It has proven to be a very secure system, and it provides a high degree of anonymity.

With the Digicash protocol a bank or financial center is able to issue *virtual cash* (a.k.a. *digital coins*) that can be sent among customers and merchants without having to reveal the identity of those exchanging them. A user can get digital cash from the bank and then she can spend it through websites that accept the type of electronic cash she got. Digicash works as follows (Figure 2).

1. The user, or *payer*, generates a random serial number on her machine that will later be turned into digital cash.

2. The payer multiplies this number by a random number raised to a public key owned by the bank that corresponds to the denomination of the digital coin desired. These denomination public keys are made available through the website of the bank so that customers can generate their own random numbers and then take them to get the bank's private signature.

3. The payer sends this number to the bank along with a payment. This can be cash, check or credit card.

4. The bank checks the identity of the customer (if necessary) and debits the payer's account with the amount requested plus a fee.

5. The bank returns the string signed with its private key for the denomination requested. Now the string has become a valid digital coin issued by the bank.

6. The payer now chooses a product to buy and she can use her digital coins to pay. She does so by sending them to the *payee* or electronic merchant.

7. The merchant can verify that they are valid coins by raising them to the corresponding public key of the bank for each denomination.

8. To verify that it is an unspent coin, the payee deposits it into her account. The bank will look at its databases to confirm the cash has not been used before.

The bank, which owns the private key of the currency signing pair, is the only one that is able to convert strings into digital coins by signing them. If someone got a copy of the signing key, she would be able to issue digital coins, which implies that these keys should be closely safeguarded. While Digicash can be anonymous, this anonymity can be lost if the customer pays for her digital coins by credit-card or check. If these payment records are exposed, it would be possible to learn the name of some of the users that got digital coins from Digicash. However, these records, and any information required to deliver the product or service purchased, are really the only evident point at which the identity of a customer could be revealed. All users, however, will remain anonymous within the group of all Digicash users. Since all coins are exactly the same (they are all random strings of the same size), the recipient of a payment cannot tell who generated the coins, but can know that they are valid unspent coins. 'After the bank has blind-signed a coin, it cannot know over what path the coins have traveled. But, in the future, it will be able to verify whatever coins it receives without knowing, however, who issued the coin initially' [7]. Once digital coins are sent to make a payment, the payer cannot be traced back by the bank.

If a user requests a large amount in *e-coins*, proof of identity will often be needed for financial and legal reasons as well. As in the real world, anonymous payments can be made to cover only reasonably small amounts of money. Remaining anonymous when handling large amounts of cash could lead to tax evasion problems, so legal limits have been imposed. If the amount of digital cash that is requested is not much, the bank can then receive cash. The strongest feature of this system is not the fact that the payer is anonymous at the moment of getting the coins, but at the moment of performing any purchase and paying by electronic cash. By controlling the amount of electronic coins being signed, Digicash makes it possible to have an auditable system that records exactly how much real cash is being received and how much e-cash is being made available to users.

At step 6 in the figure, where the payee is actually submitting the payment via digital cash, she does not have to identify herself to the payee. She just sends her money via an e-mail attachment and the payer is responsible of verifying it is legitimate digital cash (points 7 and 8). Since it is extremely easy to copy coins, all digital cash makes round trips through the banks to verify that it remains unspent. If an e-coin is sent via e-mail with no encryption and an eavesdropper intercepts the message, she can use it to go shopping.

A security improvement to Digicash would be to make sure that all users encrypt their digital cash with the public key of the merchant so that only the intended recipient could spend the money. In general, this implies reliable *public-key infrastructure* (PKI), but it is a way to guarantee that electronic cash does not get copied when crossing the network. Public-key encryption imposes a cost in transaction processing but this cost would be shared among all Digicash merchants and customers, so that its impact would be moderate.

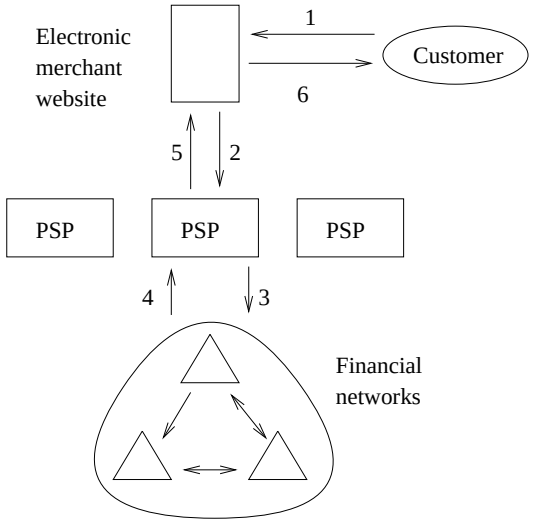

Figure 3: The Open Market system

Open Market [19]. *Open Market*(OM) is an open and extensible electronic transaction platform that is run by independent payment service providers (*PSP's*). These PSP's are the ones interacting with the banks to verify the validity of each transaction. Open Market has several peculiarities and considerable security features that make it an attractive alternative in electronic payment system on the web. This is not a centralized system where an arbiter or ruler is making sure that all transactions are valid. OM is software that is sold to people that want to receive payments over the Internet without having an external third party looking into their records. OM only makes sure that its customers are fully identified through digital certificates or any other authentication system before making the software available to them provided that, once they install an *OM server* or an *OM engine*, they are granted total trust from all other OM users. Figure 3 shows the main steps of a OM purchase transaction.

1. The user enters the merchant's website and selects a product for purchase. She provides her name and payment information so that her identity can be verified.

2. The merchant receives the information from the customer through the OM engine - installed along with the web server as a side application. It then proceeds to check if the credit or payment is valid. It does so by contacting one of the PSP's running an OM server.

3. It is the PSP the one that actually goes through the financial networks trying to complete the transfer of funds.

4. Once the payment was authorized within the financial networks, the PSP is sent an electronic receipt. This is the way to authorize that the customer can purchase the products.

5. The PSP returns to the merchant the electronic receipt indicating that the payment was good and that the products can be sent to the customer.

6. The cycle closes with the delivery of the products to the customer.

Open Market guarantees confidentiality by enforcing the use of *Secure Socket Layer* (SSL) [24] during the whole operation. That is, the customer accesses the OM engine under an encrypted SSL connection, and the merchant communicates with any of the PSP's also through SSL. Some of this traffic contains credit card numbers or information on the products being bought, but this encrypted channel prevents purchase information from being easily decoded by eavesdroppers. It is important to notice that the description of the products is not given to the PSP. Unlike other payment protocols, PSP's only get prices and product codes, but no other information about the products being purchased by a customer. Likewise, the merchant never gets any payment information typed in by the customer and the information flow is always limited to the specific entities that need access to each piece of information denying access to everyone else. The privacy of the purchase is protected to some extent since the neither the financial institutions nor the PSP's get to know what kind of products are being purchased by the customer - this information is just known by the customer and the merchant.

Notice that in step 6 an encrypted item is passed back to the merchant. This is an electronic receipt that confirms that the payment was successful. This receipt is a pair containing the identifier of the payment returned by the banking institution along with the identity of the buyer. Even when they are traveling through a SSL connection, this receipt is encrypted and hashed to protect confidentiality and integrity. It is encrypted with private key cryptography (*DES*, 56-bit keys) using short-term keys shared between the PSP and the merchant in a fashion similar to *Kerberos'* [3]. These keys are distributed using public key encryption with the keys contained in the certificates of both sides. In order to be able to be a merchant using OM, it is necessary to have a digital certificate that attaches an identity to a 1024-bit public key that is made available to all PSP's authorized to run transactions on behalf of the merchant. OM was designed to support credit card payments over the web, so it is not intended to provide anonymity - however, even when the merchant knows what products a customer is buying, she does not know the details of the payment.

Secure Electronic Transactions. *Secure Electronic Transactions* (SET) 'is a technical standard for the commerce industry developed by Visa and MasterCard as a way to facilitate secure payment card transactions over the Internet. Digital Certificates create a trust chain throughout the transaction, verifying card-holder and merchant validity, a process unparalleled by other Internet security solutions' [20]. SET is a credit-card system that relies on PKI. Because of that, its development has been slow as a security assessment of new PKI entities is necessary before granting a new participant access to the transactional services. The level of PKI present as of today is not enough to openly signal SET as a popular credit card payment system, but things have been changing lately and more support to PKI will make of this set of protocols a very secure and convenient way to submit on-line payments. A description of how SET works follows. Detailed information can be found at [7, 20] but the security features of the protocol can be seen very clearly here.

1. The customer selects a product from the website of the merchant.

2. The merchant creates an *order information* (OI) message that is returned to the costumer. It includes information such as the quantity being purchased, the code of the product and its cost. This is the initial step to verify what the user wants to buy.

3. The customer returns the OI message with her approval along with a credit card brand identifier (namely, Visa or Mastercard).

4. The merchant sends back this message including a transaction number, its key-exchange certificate, its signing certificate and a certificate of the payment gateway. The gateway is in charge of carrying out

the transaction with the financial networks (see Figure 4). The merchant signs the message with its private key of the signing pair before sending it to the customer.

5. The customer verifies the validity of the signed message by using the public key of the signing pair of the merchant. It then prepares a *payment information* (PI) message that is put along with the OI. Hash values of both of them are taken separately and then together. The customer returns this message including the key-exchange key of the payment gateway, the buyer's symmetric session key encrypted with the public half the key-exchange key of the payment gateway, the PI encrypted with that session key, and the OI and the PI together signed by the buyer.

6. The merchant verifies the signing certificate of the buyer and then the dual signature of the OI+PI packet - certificates contain the public keys of their owners. The merchant then makes an authorization request to the payment gateway by sending: The authorization request signed with the merchant's signing key, and it also includes the merchant's signing certificate, the merchant's key exchange certificate, all these encrypted in the session key that can be decrypted by the gateway. It also includes the merchant's session key encrypted with the public key of the payment gateway, the PI encrypted in a session key and the buyer's symmetric session key encrypted in the public key of the payment gateway.

7. The gateway approves the payment and sends the merchant a message that includes the response from the card issuer, identified by the brand identifier signed by the payment gateway, and encrypted with a session key.

8. Upon receipt, the merchant verifies the issuing bank response. Now the merchant actually bills the customer at the time the products ship.

9. The customer can request to the gateway that funds be moved to complete the payment. It creates a message with the OI, the transaction number signed by the merchant's signing key and encrypted with a session key as in step 7.

10. The gateway opens the message and forwards the detail to the financial institutions. They eventually move the payment to the merchant's accounts.

11. They then create a message to the merchant to confirm the payment has been done. This message is sent to the merchant once the bank signs it.

12. The merchant receives the message and updates its records with the payment information and the transaction number.

SET protocols use *dual signature* technology that involves two key pairs for each entity: A *signing pair* and an *encryption pair*. This strategy provides integrity protection and confidentiality of the purchase. Within SET, hashes of all information packets are obtained and then several packets are put together and rehashed again to provide further integrity checking. As seen above, two are the main information blocks being used in SET. They are the *Payment Information* (PI) blocks that store confidential data from the buyer (e.g., banking account numbers, credit card numbers, expiration dates), and the *Order Information* (OI) blocks that contain the specifics of the products. These two elements are put together at the end of the transaction to attach payment information to the products the customer is getting.

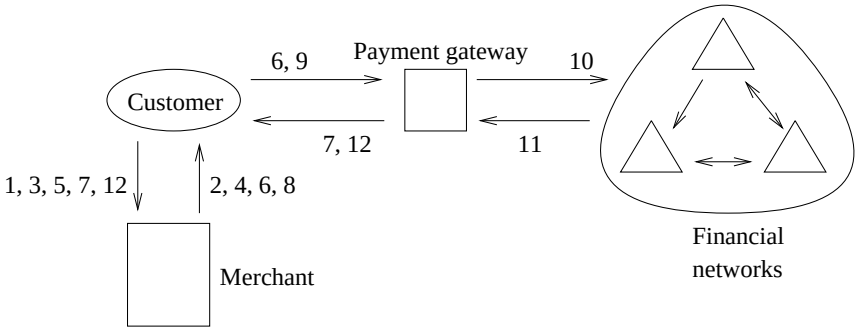

Figure 4: The SET payment model

There are some positive aspects of this protocol in terms of confidentiality and privacy. At steps 1, 2 and 3 no encryption is used at all. The information traveling between merchant and customer at this point in time is sent as plaintext since it is just the initialization step of the transaction. The information that could be obtained by intercepting these packets will only indicate that a transaction is going on between the two parties; that is, merchant and buyer are *linkable* [26]. Further steps in the protocol encrypt all the data flow to protect the information at a higher level. The use of a couple of key pairs for each entity could be thought to add a considerable overhead to the network (at least 12 keys are needed before the transaction is actually made) but, since these keys are used to exchange session private keys, performance is quite good as reported in [20].

The main security advantage SET has is a very powerful confidentiality system. Starting at step 4, all the communications travel through encrypted channels (encrypted packets) where only the legitimate recipients can get the messages. Having two key pairs per entity would make it more difficult to revoke and re-generate keys, but the benefit that is gotten from the security standpoint is worth it. Signatures and encryption are separate mechanisms that work together to provide authenticity as well. The two hashes taken at step 5 are then re-hashed together and then encrypted so integrity checking can be performed only after the recipient decrypts the packet. Another protection mechanism is encrypting the PI in a session key and sending that session key encrypted not in the merchant's key but in the payment gateway's key. This makes it impossible for the merchant to look at its content since the PI block should be only exchanged between the buyer and the payment gateway. At step 6 the information going to the gateway, both from the buyer and from the merchant, is protected by symmetric encryption with different keys, again, in a fashion similar to Kerberos ticketing [3]. Both keys are transmitted to the gateway before establishing private-key transfers. It can be seen that nothing is revealed to an eavesdropper. OI blocks are not revealed to the gateway nor to the financial networks receiving only PI blocks, which protects the confidentiality of the purchase. SET is not intended to provide anonymity and, as credit card system, it forces the use of digital certificates to authenticate the participants of a transaction.

Cybercash [21]. *Cybercash* is an electronic payment system designed to interact with personal computers. Just like Digicash, this protocol provides the convenience of storing payment items on the user's hard drive. These items can be banking account numbers, credit cards numbers and even Digicash coins. The idea of Cybercash is to have a local system running on the payer's machine that stores all these payment items so that they are available at the time the user wants to buy something on the Web. This local system runs along with the user's browser and is called electronic wallet. It works essentially as a normal wallet in the sense

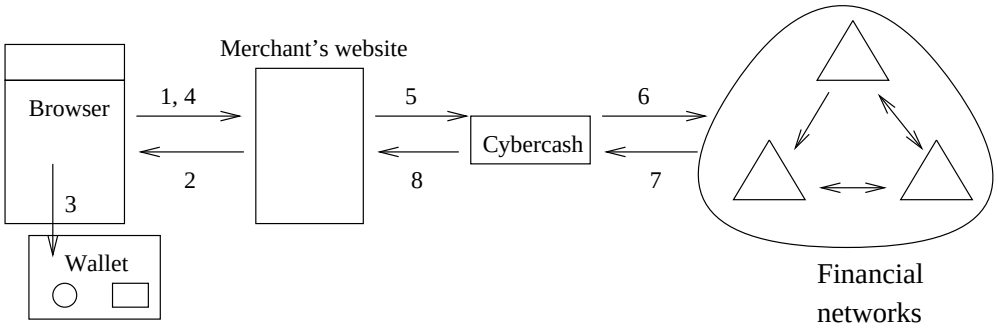

Figure 5: The Cybercash system

that the user can mix the payment items she has to complete the payment for a product. The client/server model Cybercash uses has made it one of the easiest ways to do payment verification on the web.

In order to start using an electronic wallet, each user needs to open a service account with Cybercash, just like FV does. Once the customer is debited with the amount of Cybercash payment items she wants to obtain initially, she receives an electronic wallet with her new digital coins. Then the user can open a purchase session with her browser and the wallet will be activated so that to establish connections with Cybercash servers to complete payments. The central point of any transaction with Cybercash is this electronic payment wallet. It stores confidential information and is accessed by Cybercash servers to extract the details of each payment. Here is how it works.

1. The user selects a product from a merchant's website.
2. The merchant, running a Cybercash server, receives the purchase request from the wallet of the customer and reads information from it to create the corresponding invoice. The invoice is sent to the customer through an encrypted connection.
3. The user receives the invoice and selects from her wallet the item she wants to pay with.
4. The wallet sends the payment information to the merchant using public key encryption so that an eavesdropper is not able to read it.
5. The merchant's server receives the message and encrypts it with its private key. Then it forwards it to Cybercash so that that it can verify the signature.
6. Cybercash confirms the merchant's payment request and inserts it into the financial network of the corresponding banks to execute funds transfer.
7. The financial network sends a confirmation message to the Cybercash server indicating that the payment has been successfully performed.
8. Cybercash receives the confirmation and forwards it to the merchant.
9. The products are then shipped to the customer.

Information traveling from the e-wallet to Cybercash is always encrypted using *DES* and *RSA*, but at the wallet level, no encryption is applied. The system provides password-based access but the information itself remains in binary format with no encryption. At points 4 and 5, the payment information is transmitted from the wallet to Cybercash. This information needs not to be known by the merchant, who only cares about receiving confirmation of successful payment. In other payment protocols, the binding between the identity of a user and its payment details is stored at a server. Here, that information is distributed among customers who compose a "distributed database" by storing its own personal record on behalf of Cybercash.

No anonymity is provided to a Cybercash user when getting a wallet. Opening a Cybercash service account cannot be done without showing sound proof of identity so, even when the user can send e-coins from her wallet, her identity is always known by Cybercash - but not by the merchant. A user will be able to add banking information and the credit card numbers she want to use, so the wallet will keep collecting sensitive private information. Both the merchant and Cybercash keep record of what the customer is buying and how she is paying, but the details on payments are just stored at the wallet's level that is the security point most likely to be attacked. The anonymity feature provided by this protocol is weak but it could be improved by using onion routing or anonymous protocols like Crowds and Hordes.

4 Security mechanisms and controls

As discussed in [17], there are three types of anonymous communication that can be achieved: *Sender anonymity, receiver anonymity,* and *unlinkability of server and receiver.* We now discuss the technologies that can be used to achieve these different levels of anonymity as required by each System.

Chaum's mixes. *Chaum's digital mixes* [22] are the foundation of many anonymizing services that are currently available. They consist of a set of forwarding nodes that exchange encrypted messages. Public keys for all forwarding nodes are made available to senders so that random transmission paths can be created. All packets are the same size so that size does not help reveal the identity of a sender. Dummy messages are also sent from all the nodes of a *mix level* to each recipient of the next level.

The originator of a message uses the public key of each mix in a randomly generated path to compose the message and send the message to the first level. The first mix level node decrypts the message with its private key so that it can see the address of the receiver at the next level. It then forwards the message to the next level, and so on. From level 2 and above, a forwarding node cannot know who the originator of the message is. The receiver will finally receive a message from the last mix level but will never find out who is behind the mixing network. Replies are sent along the reverse path the message traveled on the way to the receiver.

Onion routing is a technology based on mixes and can be used in electronic payment systems. Figure 6 helps explain how digital coins can be sent to a recipient using onion routing. The customer submitting a payment creates a message that contains some digital coins. This message is signed and encrypted with the public keys of the members of a randomly choosing forwarding set that compose the path between payer and merchant. The high degree of anonymity onion routing provides makes it a viable method of improving anonymity of electronic cash protocols. Since the number of packets involved in a digital cash transaction is not large, the performance cost the use of public key encryption imposes is small and could be overlooked by merchants that only deal with a small number of customers.

Crowds[16, 17]. *Crowds* is the name of a randomized routing protocol that has been developed to provide

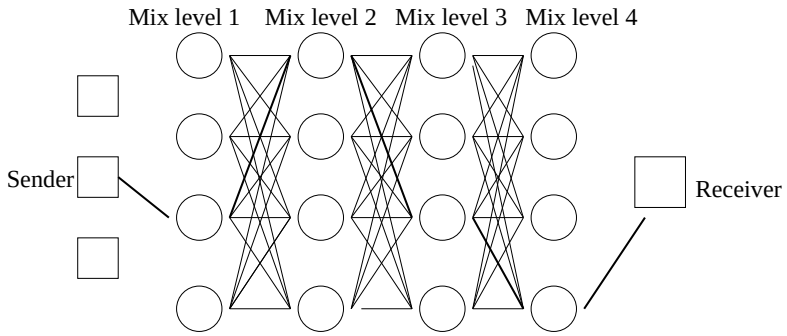

Figure 6: Chaum's mix

anonymity on the Internet. The main idea of this protocol is to create user groups in which members forward messages among themselves before sending it to the final receiver or responder. This way the identity of the originator remains hidden. Each user member of a crowd has a proxy process called *jondo* that communicates with the other jondos when forwarding messages. First the originator sends the message to a random member of the crowd. This member, based on a *forwarding probability, pf* forwards the message to the receiver or to another randomly chosen member of the crowd. The message travels across the crowd before being delivered to the intended recipient.

At any point in time, a jondo cannot tell who the initiator of a message is. Notice, however, that the identity of the recipient needs to be known by the final jondo that will forward the message to the recipient. Crowds can be used to provide strong anonymity for electronic cash systems and electronic wallets. The messages that would be exchanged between customer and merchant (sender and responder, respectively) are electronic coins to perform a payment, and the payment receipt as response to the buyer. The use of Crowds to provide anonymity to payment systems has some advantages. There is not a single repository where the identities of crowd members are stored [17] and payments are anonymous all the time as they crosses the network.

Hordes [17]. *Hordes* is a protocol that provides receiver anonymity over the Internet. It uses multicast routing to reduce the latency of data delivery, being this one of its main benefits, and could be used to improve anonymous electronic payment systems where response time is a concern.

It could be used in the following manner to transmit digital cash. The initiator of a communication sends messages through a path made up of proxies or jondos that are randomly chosen. The originator then randomly picks a jondo and sends the message to it. When sending a message, as in Crowds, a forwarding probability dictates whether a jondo forwards the payment message to another jondo or if it sends it directly to the final receiver, which could be a merchant or financial institution. Once digital coins are sent, a return receipt needs to be sent to the originator. Hordes uses the anonymity inherent to multicast point-to-multipoint IP routing. In order for Hordes to be able to send a reply (return receipt), the initial message sent by the originator using randomized forwarding should always include a multicast address the originator. In that way the response can be sent directly to that address using UDP-encapsulated packets. See Figure 7 for an overall description of how Hordes works.

Even if the membership of a multicast group were exposed, the sender of a payment would still remain anonymous within the group of members of the multicast group. Since no routing information is stored at the forwarding jondos, Hordes avoids a passive trace back attack [17] so any digital coin sent through it

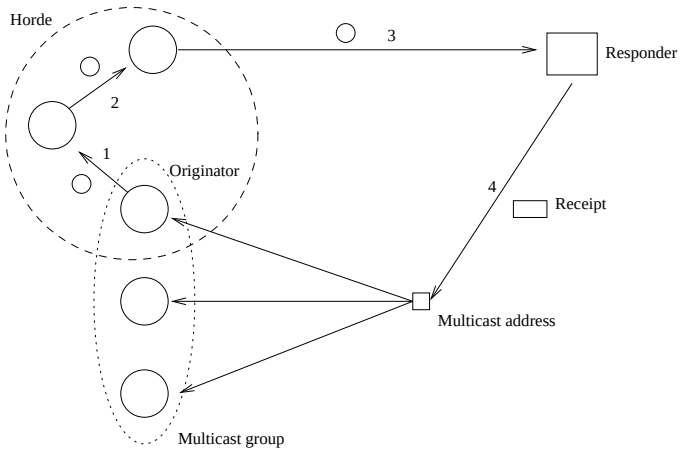

Figure 7: Sending digital coins through the Hordes protocol

could not be monitored. Hordes and Crowds are excellent means to provide anonymity for payment protocols at the cost of delivery latency, but, since performance is not the most important concern for this kind of systems, they could be used.

5 Conclusions and remarks

The main advantage of cash systems is anonymity — the user does not need to give proof of identity when making a purchase. Most of the current e-cash protocols, however, lack strong anonymity capabilities that could effectively hide the identity of their customers. These systems could be strengthened in two ways. Anonymity could be designed into new protocols, or it could be added by using existing and proposed protocols for anonymous communication. A level of indirection can be used in credit card-based systems to substitute credit card numbers by other sort of numbers so that sensitive information does not travel across the communication channel every time a payment is made. Even when this idea goes against the convenience of using actual credit cards to shop on the Web, it guarantees a higher lever of security.

Chaum's mix-based systems help improve the level of anonymity a system provides. For payment schemes where response time is a concern, protocols like Hordes could be used to enhance the quality of the anonymity provided to the users improving delivery latency at the same time. Public Key Infrastructure is extremely important and will become even more important to digital cash protocols as they rely on electronic signatures to issue coins and authenticate customers. The success of all novel digital cash system will very likely depend on the support and quality of this infrastructure.

References

[1] A. Ambross. *USA Today.* Computers section. 26/June/2000.

[2] B. Marat *Privacy Times,* www.privacytimes.com . 11/June/2000.

[3] S. Garfinkel and E. Spafford. *Practical UNIX and Internet Security* O'Reilly and Associates. 1996.

[4] B. Richard. *Computer Security Handbook* . 1993.

[5] L. Detweiler. *Identity, privacy and anonymity on the Internet* . 1993.

[6] C. Pfleeger. *Security in Computing.* 2nd Edition. Prentice-Hall. 1997.

[7] S. Garfinkel and E. Spafford. *Web Security and Commerce.* O'Reilly and Associates. 1997.

[8] A. Rubin, D. Geer and M. Ranum. *Web Security Sourcebook.* Wiley Computer Pub. 1997.

[9] V. Ahuja. *Secure Commerce on the Internet.* AP Professional. 1997.

[10] First Virtual. www.fv.com . June/2000.

[11] R. Harrison. *ASP/MTS/ADSI Web Security.* Prentice Hall PTR . 1999.

[12] K. Lamnd. *The Virtual School,* www.virtualschool.edu/mon/ElectronicProperty/klamond . November/1996.

[13] P. Lewis. *Ignite Debate on Anonymity*, The New York Times. 31/December/1994

[14] K. Ivey. *Cookies: Just a Little Data Snack.* The Editorial Eye, www.eeicommunications.com . February/1998.

[15] D. Chaum. *Security without identification: transaction systems to make big brother obsolete.* Communications of the ACM. October/1985.

[16] M. Reiter and A. Rubin. *Crowds: Anonymity for Web Transactions.* ATT Labs-Research.

[17] C. Shields and B.N. Levine. *A Protocol for Anonymous Communication Over the Internet.* To appear in Proceedings of the ACM Conference on Computer and Communication Security 2000. November/2000.

[18] DigiCash (E-Cash), www.digicash.com 1999.

[19] Open Market, www.openmarket.com . 1999.

[20] Secure Electronic Transaction-SET, www.setco.org .

[21] Cybercash, www.cybercash.com .

[22] D. Chaum. *Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms* Communications of the ACM. February/1981.

[23] D. Chaum. *Security without Identification: Transaction Systems to Make Bib Brother Obsolete.* Communications of the ACM. October/1985.

[24] K. Hickman T. Elgamal. *The SSL protocol.* Internet draft (draft-hickman-netscape-ssl.01.txt)

[25] M. Reiter and A. Rubin. *Crowds: Anonymity for web transactions.* ATT Labs - Research.

[26] P. Syverson and S. Stubblebine. *Group Principals and the Formalization of Anonymity.* FM 1999.

[27] D. Atkins. *Magic Money.* www.csn.org/pub/mpj/crypto/mm.htm

[28] Pay Pal, www.paypal.com . July/2000.

[29] Ecount, www.ecount.com . February/2000.

[30] PyByCheck www.paybycheck.com . September/2000.