A SURVEY OF PEER-TO-PEER MICROPAYMENT SCHEMES

Mohit Jain, Siddhartha Lal and Anish Mathuria
Dhirubhai Ambani Institute of Information and Communication Technology (DA-IICT)
Gandhinagar, Gujarat, India – 382 007
{miohit_jain, siddhartha_lal, anish_mathuria}@daiict.ac.in

Abstract: P2P networks suffer from the free-riding problem, wherein users connect to the network only to use the resources, but do not donate any resources themselves. A popular approach to overcome this problem is to reward peers who share their resources, through payments. This has given a fresh lease of life to micropayments schemes, which are now being used in such networks. P2P Micropayment schemes need to address several requirements like transferability, anonymity, etc. that are not met by many traditional micropayment schemes. They must utilize the distinctive features of a P2P network to maintain high efficiency, while providing optimum security. In this paper we list out the requirements that need to be kept in mind while designing such schemes, present a survey of existing schemes, and analyze them critically. We also give a comparison of the schemes with respect to security and performance.

Keywords: Peer-to-Peer; P2P; Micropayment; Electronic Commerce; Survey; Comparative Analysis; Requirements; Double Spending Detection; Double Spending Prevention; Fair Exchange; PPay; Karma; FairPeers; P2P-NetPay; File Market

1 Introduction

The availability of low cost digital content on the Internet triggered the advent of micropayment schemes during the 1990s. These schemes were the outcome of a change in the mindset of Internet content providers. In the past, content was provided for free by altruistic individuals or organizations such as universities. This, many a times, led to copyright infringement and lack of accountability. With the phenomenal growth of the World Wide Web, content providers began to seek profit for their services. Online advertising annoyed users, giving way to schemes allowing *pay per view*. However, existing e-commerce payment schemes (like credit or debit cards) were not suitable for handling low valued payments as the processing cost usually exceeded the payment (Wikipedia, 2008). Also, these payment methodologies involved substantial delay, user involvement, and potential for disputes (Kou, 2003). This led to substantial research in the field of Micropayments and several such schemes were proposed. Increased interest in selling intellectual property over the World Wide Web also supported micropayments.

Micropayments refer to payments so small that processing them through conventional channels is relatively costly. A credit card payment involves a processing fee of about 15 to 35 cents (Geer, 2004; Wei et al., 2006). Hence going by this definition, micropayments are payments of value less than or equal to 50 cents. Most of the micropayment schemes make use of electronic coins (hash chains or signed messages) which are aggregated over a period of time and then presented to a trusted authority (broker or bank) for redemption. Security is relatively unimportant and fraudulent users are deterred by making cheating detectable, traceable, and unprofitable. Therefore, micropayments mostly seek to provide optimum security while maintaining high efficiency.

In recent years, several arguments have been made against micropayment methodologies, most notably (Odlyzko, 2003; Someren et al., 2003). Most of these arguments have had a sociological, economical, and psychological basis. Odlyzko (2003) argues that factors like competition from other payment schemes (namely, existing options like credit and debit cards), long incubation period for any new economically feasible technology, behavioral economics, advantages of aggregation strategies, and reluctance of government agencies and service providers to respect individual privacy will forever restrict micropayments to a marginal role in the economy. It is true that several micropayment startups have appeared and more are likely to appear in the future. However, it is circumspect whether venture capitalists will continue investing in a prospect that doesn't seem to guarantee *immediate* returns. More recently, however, micropayments have gained a fresh lease of life with the advent of Peer2Peer (P2P) networks. Several commercial P2P applications are being launched and there are talks of big corporate houses collaborating with existing networks (Warner Bros, 2008). P2P applications have become a powerful means to share colossal volumes of data, computational resources, intellectual property, and other resources. There's an increasing interest in implementing micropayments in existing P2P networks (Yang and Garcia-Molina, 2003). There are several strong reasons for doing so.

Firstly, P2P networks usually involve sharing of low cost content (like mp3 files or divx video clips) and by providing incentives (through micropayments) to peers who share their resources, the well known free-riding problem (users connecting to the network only to use the resources, but do not donate any resources themselves), can be reduced considerably. Secondly, researchers have argued that micropayments can be used to impart fairness to a profit sharing environment wherein the rights of the original owner can be protected (Catalano and Ruffo, 2004). Thirdly, up till now, P2P networks have not been commercially viable because of copyright issues. By incorporating micropayments P2P networks can gain wider acceptance (EMI Music, 2008; Warner Bros, 2008). Fourthly, P2P networks are usually self-organized and robust, mostly independent of centralized servers, scalable, and rich in resources at the edge of the network (Zuo and Li, 2005). These are some of their most inviting features and can help to make micropayments scalable and economically feasible in real world applications.

A large number of existing micro-payment schemes like Millicent (Glassman et al., 2008), PayWord (Rivest and Shamir, 1997), etc. are not well suited to P2P. Millicent requires an online broker to check for all the transactions, depriving the system of scalability, whereas scalability is one of the most attractive features that a P2P system offers. The payword coins are inherently untransferable.

In this paper, we give a brief overview of the requirements (Section 2) that need to be kept in mind while designing P2P micropayment schemes. We also present a survey of existing schemes in Section 3. In Section 4, we draw a comparison chart of the studied protocols on the basis of certain performance metrics. We also highlight some of the issues that might help future protocol designers in Section 5. To the best of our knowledge, no such survey on P2P Micropayments exists and it is hoped that this study will benefit and assist researchers who are new to the field.

2 Requirements

As discussed previously, micropayments are payments of small amounts (usually less than 20 cents) that are made electronically. Since it is essential that the cost of the scheme doesn't exceed the face value of the payment, there are certain requirements that need to be kept in mind while designing such schemes. Usual prerequisites like optimum security, detection of fraud, and trusted authority's load are important factors. However, in the P2P scenario, there are some additional requirements which need to be adhered to. These ensure that the unique characteristics of P2P networks are utilized to maximize efficiency and security. These requirements can be roughly divided into two main categories - mandatory and desired.

2.1 Mandatory Requirements

The mandatory requirements listed in the order of decreasing importance:

2.1.1 Transferability

In a P2P network, peers can be consumers as well as vendors. Any scheme should therefore allow payments (electronic coin, electronic lottery ticket, electronic cheque) to be transferred from one peer to another without the involvement of a Trusted Authority. In most schemes, transferability is implemented by adding a layer (containing peer specific information) to the coin when it is passed onto another peer. This increases the time it takes to detect a double spending fraud (refer Section 2.1.2), while also increasing the size of the coins (Chaum and Pedersen, 1992). However, Yang and Garcia-Molina (2003) makes use of peers in order to implement transferable coins that do not increase in size.

2.1.2 Double Spending Detection

Any electronic payment mechanism needs to make sure that fraud is computationally infeasible. Most micropayments schemes (to the best of our knowledge), however, strike a tradeoff between utmost security and efficiency. They deal with double spending by making it detectable, traceable and unprofitable. Therefore, if a user tries to commit fraud, then it is ensured that he will be shunned from the system. The risks involved in committing fraud far outweigh the benefits.

It should be noted here that any scheme can be made foolproof by making use of a Trusted Authority for all transactions. However, such a scheme would be highly inefficient as the broker load will be *O (n)*, where *n* is the number of transactions. Offline payments are preferred because they have *lower latency, communication costs and computational costs* (Yang and Garcia-Molina, 2003).

2.1.3 Scalability

Most conventional micropayment schemes make use of a Trusted Authority (T for convenience) to eventually validate all the transactions that take place between the consumer and the vendor. Thus the T load is always *O (n)* where n is the number of transactions. In such schemes, the Trusted Authority becomes the scalability and performance bottleneck (Yang and Garcia-Molina, 2003), as well as a single point of failure. However, P2P networks achieve remarkable scale because of their inherent capacity to exploit huge amounts of resources at the edge of the networks (i.e., peers). Thus it is necessary for a P2P micropayment scheme to attain scale by making use of these "edge" resources and spreading the load of the trusted T over several untrusted peers.

2.1.4 Offline Trusted Authority

Since detection of fraud, auditing, and banning of fraudulent users is ultimately the responsibility of the Trusted Authority (T for convenience), it can not be ruled out completely. Most protocols try to minimize its load, which automatically enhances the efficiency of the scheme. With the exception of few schemes like Payword, in most of the micropayment schemes, the transactions are usually online and involve the arbitration of T. P2P micropayments, however, can make use peers in order to make these transactions offline and involve T only after a payment has been transferred substantial number of times or when several payments have aggregated. The security issues are dealt with by the peers themselves by ensuring that the protocol makes fraud *computationally infeasible*.

2.2 Additional Requirements

2.2.1 Anonymity

One of the most striking features of micropayments was their ability to ensure anonymity of online monetary transactions. Spender's are in favor of anonymous transactions, just like the cash payments. However, experience with schemes providing anonymity (Jia et al., 2005; Wei et al., 2006) has suggested that the benefits are often outnumbered by the shortcomings. A lot of resources are utilized in ensuring untraceable transactions which results in the cost of the scheme exceeding the value of the payment. Moreover, due to sociological reasons such as money laundering, tax evasion, and funding of terrorist activities, reluctance of government agencies to protect individual privacy has significantly reduced the role of such protocols in economy (Odlyzko, 2003). Sellers are also against anonymity, as it helps in selling goods as per the customer's choice (Lesk, 2004). Thus, although anonymity is a desirable requirement in some P2P applications, it remains to be seen whether they are sustainable in the real world.

2.2.2 Double Spending Prevention

Double spending prevention, as the term implies, refers to preventing coin fraud at the outset. As noted by Hoepman (2006), this apparently means that all transactions need to be online and through a trusted authority. The load of the Trusted Authority is increased considerably which hampers the efficiency of the scheme, although better security is ensured. Although the overhead involved in implementing such a feature is not negligible even in P2P networks, some schemes like Vishnumurthy, Chandrakumar, and Sirer (2003) have tried to incorporate it. It should be kept in mind, however, that micropayments were originally meant to provide high efficiency while guaranteeing optimum security. So the feasibility of such schemes remains circumspect.

2.2.3 Fair Exchange

Wherever money is involved (like commercial P2P applications), trust is a very important issue as peers are likely to cheat one another. In order to deter, it is essential that some mechanism is in place in order to ensure fair exchange of goods. By fair exchange we mean that a peer receives service if and only if he pays for it. No honest party suffers a loss of any significant value. There are several protocols in literature for implementing optimistic fair exchange (Micali, 2003) and some of the schemes (Zuo and Li, 2005) that we studied make use of them in order to ensure fair exchange of goods in a P2P market.

2.2.4 Fairness

In some P2P applications it is essential that both, the owner and distributor of a file, get credit for sharing their resources. Copyright should not be violated and the author or owner should get his due, even if he is not directly involved in the transaction from one peer to another (Catalano and Ruffo, 2004). This requirement is different than Fair-Exchange (discussed in Section 2.2.3) and involves protecting the intellectual property rights of the original owner of a file (for instance an mp3 audio clip or an animated video). Emerging commercial P2P applications will need to cater to this requirement as copyright infringement is a major issue, for example, the legal action taken against Napster (Napster, 2008) (which had so far kept P2P networks out of the legal business).

3 Existing Schemes

In this section we give a brief overview of the major schemes that we covered in our survey. One needs to keep in mind that this is not a comprehensive list and covers only those protocols which provide for some unique features. Also, for the schemes using asymmetric key encryption, it is assumed that each user knows the correct public key of every other user.

3.1 PPay

PPay (Yang and Garcia-Molina, 2003) is one of the pioneering works in the field of P2P micropayments. Several other schemes such as FairPeers (Catalano and Ruffo, 2004) and WhoPay (Wei et al., 2006) are based on PPay. It allows coins to be transferred from one peer to another without the involvement of a trusted authority, T. T participates in transactions in which the coin is created or redeemed and hence its load is *O(n)* in the number of coins. PPay coins are self-managed by the node who "owns" the coin and all the security issues related to a coin – like double spending detection, forgery prevention – are handled by the owner of the coin. Double spending is possible but it is made unprofitable by ensuring traceability of the malicious peer. In the following, we present a basic version of the PPay protocol. For a detailed discussion of the security issues or extensions to the scheme, please refer to (Yang and Garcia-Molina, 2003).

In the basic PPay scheme, each user (say X) buys a certain number of coins from the trusted authority T. The coins are messages of the form C = {X, sn}, which are signed by T for authenticity. X becomes the owner of the coin C. The serial number sn is unique for each coin. A coin is of a fixed denomination. In order to pay for a service, the owner X assigns coin C to the vendor Y by signing the message A_{XY} = {Y, seq_1, C} with its private key (Step 2.1 of Figure 1), where seq_1 represents the sequence number of the assignment. Any new assignment of the coin C must have a higher sequence number than the previous assignment(s), as sequence number is used to detect double spending. Y now becomes the holder of the coin. In case Y needs to spend this coin at another peer Z, it sends a reassignment request to X (Step 3.1 of Figure 1). The owner X now reassigns the coin to Z by signing the message {Z, seq_2, C}, and also sends a copy of the reassigned coin to Y. Here, seq_2 is greater than seq_1. Due to the reassignment of the coin C, the previous assignment of coin C by X to Y is no longer valid. X keeps an audit trail of all the coins owned by it as the onus of proving the invalidity of a transaction lies on the owner and not the holder of a coin. If X is offline, then Y sends the reassignment request to the broker. B generates the newly assigned coin, identical to the assignment issued by the owner X: A_{XZ} = {Z, seq_2, C}$_{SKB.}$ When X comes back online again, B sends the reassignment request by V to U. In this case, the broker load is *O(# of transactions)*. It has the disadvantage that the broker is kept online. Moreover, the broker has to maintain an audit trail of all the coins it issues while the owner has to do the same for all the coins it owns or holds.

Yang and Garcia-Molina (2003) also proposes an alternative to the basic PPay scheme using layered coin architecture, wherein Y can avoid the reassignment phase (through the owner X of the coin) and assign the coin to another user Z by signing the message {Z, Y, seq_n, A_{XY}} with its private key. This layering of coin serves as a proof that the holder has relinquished its hold on the coin. When a node eventually approaches the original owner X of the coin, it can obtain all the necessary information by peeling off these layers. Layered coins do not offer anonymity, are not untraceable, and grow in size. Layering, even though more efficient than the reassignment process, delays the detection of coin fraud as Y can replicate the coin without anyone's knowledge. The fraud is detected only when the original user X peels off the layers.

3.1.1 Security

Forgery Prevention:

As each digital coin is signed by T, thus no malicious peer can forge a coin, assuming digital signatures are not forgeable. Also, reassigned coins are signed either by the owner of the coin or T, in case the owner is offline, preventing forging.

Double Spending Detection:

There are three cases to consider:

First, the owner X of coin C may commit fraud by assigning the same coin to both A and B. In this case, the double spending is detected at T during redemption, as T keeps a record of the serial number, owner of each redeemed coin and the holder of the coin who redeemed it. If T discovers that multiple coins with the same serial number and owner are redeemed, T knows either the owner or the holder of the coin is the cheater. When enquired by T, if X refutes one of the assignment, then the holder of the coin is punished, else the owner.

Second, the holder Y of a coin C may try to double spend the coin by assigning the same coin to two peers A and B. However, owner X will refute the second reassignment request of coin C.

Third, let the owner X of coin C, assigns the coin to Y, and Y reassigns C to Z. If both Y and Z ask T for redemption of coin C, T checks the sequence number on the assigned coins with same serial number and who ever possess the coin with a larger sequence number is the true holder of the coin and the other peer is punished by T.

3.2 WhoPay

WhoPay (Wei et al., 2006) is an extension of PPay that provides anonymity. As in PPay, the peer (say X) buys coins from the trusted authority T, and becomes the owner of the coin C (Step 1.2 in Figure 2). WhoPay uses group signatures (Chaum and Heyst, 1991) to provide anonymity and to reveal the identity of a node in case double spending is detected. When a user X joins the WhoPay system, it needs to register with a trusted authority, called the judge. The judge assigns each user a distinct private group key (gk_X). (Note: In WhoPay, all users belong to the same group). A malicious node's identity can be revealed by the joint effort of Judge (who holds the master private key) and T.

Any transaction in WhoPay requires the vendor, say Y, to generate a new public-private key pair (pk_{CY}, sk_{CY}). Instead of representing coins by a serial number (as in PPay), coins are represented by a public key. The peer who knows the corresponding private key is the holder of the coin. For a transaction, owner X signs the coin C, along with pk_{CY} and sequence number seq_1, and assigns the coin to Y (Step 2.1 in Figure 2). Now Y becomes the holder of the coin C, as the corresponding private key sk_{CY} is only known to Y.

In WhoPay, owners of the coin use only their private key (here sk_X) to sign the messages (Step 2.1 and Step 3.2). However, any message from a coin holder, say Y, is signed using two keys (step 3.1 – request for reassignment of coin) – the coin's private key, sk_{CY} (which proves the holdership of the coin) and the peer's group private key, gk_Y (which is used to identify the malicious party in case a fraud is detected). Thus, WhoPay provides full anonymity to the holder of a coin, but no anonymity to the coin's owner. During a transaction between peers Y and Z, neither the payer nor the payee identity is revealed.

WhoPay has some obvious disadvantages that are inherited from PPay. Assuming the owner is usually offline, the load of the broker increases to *O(# of transactions).* Generation of a public-private key pair for each of the transactions is very resource intensive and also requires the peer to remember each such pair for all the coins it holds or owns.

3.2.1 Security

Forgery Prevention:

WhoPay is as secure as PPay, as all the coins are signed by T and during reassignment the coins are either signed by the owner of the coin or T, thus preventing forging of coins.

Double Spending Detection:

Anonymous transactions make double spending detection more complex. To detect double spending in real-time, WhoPay proposes a novel approach based on Distributed Hash Table (DHT). The idea is to publish a globally readable list which binds coins to the peers. In such a case, a vendor accepts the payment only after verifying that the coin is binded with the newly generated transactional public key. The list can only be modified by the owner of the coin or T (when the owner is offline), but the list is universally readable. Each peer can monitor the public bindings for the coins it currently holds using polling or register/notify mechanism, and any unexpected update is reported to T. T and Judge jointly detects the fraud and malicious node is punished.

3.3 FairPeers

The FairPeers protocol (Catalano and Ruffo, 2004) is another protocol based on PPay (Yang and Garcia-Molina, 2003). It aims to achieve copyright protection by transferring the selling rights of a digital content from peer to peer. To that end, the scheme introduces a new trusted entity called Copyright Granter (CG) that generates a certificate (CC_F) binding a file F (with metadata F' and life span LS_F) to its author A (Step 1 of Figure 3).

For every transaction, the customer Y needs to pay both – the author A (the peer holding the copyright) of the file F and the vendor X of the file, using two different assigned coins, CYX and CYA (Step 3 in Figure 3). If peer X buys the file F directly from the author A, two coins (CXA and C'XA) needs to be assigned to the author A (Step 2 of Figure 3). If the author is offline, then similar to PPay downtime protocol (discussed in Section 3.1), CG takes place of the author for that transaction and pays back the author when it is online again. For the format of each message of the protocol for the transaction, please refer to (Catalano and Ruffo, 2004).

Catalano and Ruffo (2004) also propose an interesting approach of reassigning coins by means of delegation of accountability. When a peer X (re)assigns a coin to a peer Y, along with the coin, X also gives Y the right of reassigning the coin to other peers in the form of a delegation token. A delegation token D_n consists of:

$$D_n = \{C_{XY}, LS, PK_X\ PK_Y^{del}, T, D_{n-1}\}_{SKx}$$

where X, the n^th owner of the coin C_{XY}, is assigning the coin to Y, the (n+1)^th owner. LS is the lifespan of the coin, PK_X and PK_Y^{del} are the public keys of X and Y, respectively. For every reassignment, the receiver (here Y) generates a new delegation key pair (PK_Y^{del}, SK_Y^{del}) that will be used to further reassign coin C_{XY} to other peers. T is the transferability bit, whose value determines whether the coin can be reassigned only or redeemed only. As D_n contains the full trace of the coin assignment, any kind of fraud can be easily detected. But this approach suffers from the similar problems of layering coins, as the size of D_n increases after each reassignment.

FairPeers protocol is limited to one-to-one transactions – a source file can only be downloaded by one user at a time – a quiet unrealistic scenario. Ruffo and Schifanella (2007) proposed an enhanced version of FairPeers protocol which is independent of PPay and allows multiple source downloads.

3.3.1 Security

Forgery Prevention:

Similar to PPay, in FairPeers each coin is signed by the user assigning (or reassigning) the coin, thus preventing forgery of coins. Also, a user X cannot claim to be the author of a content it did not produced, because it needs the copyright certificate signed by CG to prove this fact, thus the scheme prevents forgery of content too.

Double Spending Detection:

In reassigning coins by means of delegation of accountability, the delegation token D_n contains the full trace of the coin assignment. Thus, when a user ask the trusted authority T for redemption of coin, T checks for any kind of double spending fraud (similar to layered coins architecture) and punishes the culprit, if any.

3.4 Karma

Karma (Vishnumurthy, Chandrakumar, and Sirer, 2003) is the first fully decentralized P2P micropayment scheme; unlike previous schemes it doesn’t require a trusted authority. The overall standing of each peer is represented by a single scalar value, called *karma* of that peer. Thus in the Karma scheme, digital coins are replaced by *karma*. Each peer (say X) is associated with a bank-set ($bank_x$) consists of a set of nodes (peers) (see Figure 4). The bank-set represents a semi Trusted Authority. In Karma scheme, the bank set of a peer keeps track of the resources consumed and contributed by that peer.

Any transaction between the peers involves the peers’ bank-sets interacting with each other to transfer Karma (Steps 2 and 3 of Figure 4). All the nodes in the bank set of the buyer (say X) send messages to all the nodes in the bank set of the vendor (say Y). Thus, each transaction involves an exchange of $O(n^2)$ messages, where n is the size of the bank set. As there is no Trusted Authority, each decision is taken by a majority voting, assuming a majority fraction of the bank set is non-malicious. Once the exchange has been confirmed by $bank_Y$ (Step 5 and 6 of Figure 4), Y provides the desired service to X (Step 8 of Figure 4). Vishnumurthy, Chandrakumar, and Sirer (2003) provide a high level description of the protocol flows, the format of messages is undefined.

This scheme involves the exchange of a lot of messages and heavy computational overhead. However, this overhead utilizes the resources of the peers and thus, supposedly, maintains efficiency. Moreover, in case the nodes in the bank set of a peer are awarded any karma for their resources (used in the exchange for karma), the bank-sets can probably land up in an infinite process.

3.4.1 Security

Forgery Prevention:

In Karma, the money (or karma) of each peer is with the peer’s bank-set, and any transfer of money involves the bank set, thus avoiding forgery.

Double Spending Prevention:

Karma claims higher security by providing double spending prevention, unlike previous schemes providing double spending detection. Each transaction in Karma, involves bank-sets of the spender and vendor. As the bank-sets takes decision through a majority voting, assuming a majority fraction of the bank set is non-malicious, thus prevents double spending.

3.5 Off-line Karma

Based on Karma (Vishnumurthy, Chandrakumar, and Sirer, 2003), Garcia proposed Off-line Karma (Garcia and Hoepman, 2005), claiming a complete decentralized efficient P2P micropayment system that detects double spending. In this scheme, the owner (or buyer) mints coins and transfer it using the layered coin architecture (discussed in Section 3.1). Coins are minted by finding collisions on a hash function, an expensive but feasible operation. To generate a coin, a user *U* finds a collision satisfying: $h_1(x) = h_2(y)$ and $x \neq y$, where: $x = u||sn||ts$, h_1 and h_2 are hash functions such that $h_1 : A \rightarrow C$ and $h_2 : B \rightarrow C$. The minted coin contains the user’s identity, serial number *sn* and time stamp ts. User identity and *sn* constitute the unique coin identity. The new karma coin is defined as: $k_0 = <x, y>$.

To purchase an item, the buyer U endorses the karma coin $k_{i+1} = \{k_i, z, V, C_u\}K_u$ to the vendor V

(Step 1 of Figure 5), including vendor's identifier, *U*'s certificate C_u allowing *U* to mint coins and random challenge *z* generated by the vendor. Transactions between buyer and vendor do not require a third party. Instead, to detect fraud and to reduce size of the layered coin, karma coins need to be reminted occasionally.

Off-line Karma replaces the centralized role of bank for reminting the coins by reminters set. Each peer *V* is associated with a neighbour set $N_r(V)$, consisting of the first *r* on-line nodes close to V. In order to remint coin *k*, user *V* sends *k* to each node in the remint set, $R_k = N_r(h(id(k)))$. The reminter set must consist of at least one non-corrupted node to detect double spending. This condition is arguable, as the honest reminter can be treated as a broker leading to a centralized system. The new reminted coin is $k_{new} = \{k_0, ts, R_k, C_{Rk}, V\}R_k$, where *ts* is the timestamp and C_{Rk} is certificate of the reminters set R_k (Step 2 of Figure 5). As the reminted coin contains the multisignature of the reminters set, thus even after reminting, the size of the new coin remains considerable.

Decentralized reminting requires several message passing, leading to high overhead. Moreover, each reminter node needs to perform the same computation to check for double spending, leading to wastage of resources of the system. Reminting is against a peer's interest as it requires the peer to send the coin to each node in the reminter set. Hence in a real world scenario, reminting will be asked by a peer only at the end of *time-to-live* of the coin. This difference in time between a coin is minted and reminted is directly proportional to the size of the coin and more importantly, to the time to detect frauds.

3.5.1 Security

Forgery Prevention:

For any transaction, the reassigned coin contains the vendor's identity and is signed by the spender (the peer currently holding the coin), thus preventing forgery.

Double Spending Detection:

Due to the layered coin architecture, each coin has a sequence of signatures which records the payment history of that coin. During reminting a coin, if the reminters set obtain two coins with the same coin identity, then the payment history of the coins are compared, and the cheater is punished. Thus, along with reducing the size of the coin, reminting also detects double spending.

3.6 Fair Exchange File Market

This scheme (Zuo and Li, 2005) borrows ideas from Micali (2003) in order to upgrade an existing P2P network to a file market wherein peers pay to download a file. This is essentially done in order to discourage free riders. The payment is in the form of a virtual currency (signed message by the payee) which is redeemed by a trusted authority (T for convenience, see Figure 6). Unlike other P2P micropayments schemes, the payment cheques are non-transferable and must be deposited with T after a sufficient number have been collected. There is a fee associated with this redemption, in order to deter peers from approaching T after every transaction. The scheme claims to provide fair exchange by means Trusted Third Party (TTP), who is involved only in case of dispute. The scheme is, thus, optimistic in this regard.

Each user is assigned a CC from the Accounting Centre (AC). AC acts as the central authority and is also responsible for issuing public private key pair to each peer when it joins the network. CC is a digital certificate that states the maximum amount a downloader can pay using cheques. The Fair Exchange File Market Protocol basically consists of three phases (the third phase occurs only in case of disputes). The first step is the Negotiation Phase, in which the downloader X sends the signed download request REQ to the provider Y, along with Capital Certificate CC proving that it has enough money to pay for the requested goods (Step 1.1 of Figure 6). The download request REQ consists of the id of the file which X wants to download, X's identity, serial number of the cheque (which X promises to give to Y after receiving the requested goods) and the timestamp TS. After receiving the message 1.1, Y verifies the signature on the download request and the capital certificate. If Y is satisfied, it encrypts one of the pieces of the requested file, and sends its sequence number to X else sends an error message (Step 1.2 of Figure 6).

The second phase is the Payment Phase. After getting all the pieces of the file (including the encrypted file piece, $SE_K(KP)$), Y signs the cheque and sends it to be signed by X (Step 2.2 of Figure 6). The cheque information consists of two parts: C and Z. C contains the identity of the provider Y, the downloader X, and the trusted third party T, along with the serial number of the cheque (chequeSN), the amount to be paid, time stamp TS, the hash of the key piece, and the hash of the encrypted key piece. The second part Z, consists of the symmetric key K (used to decrypt the encrypted file piece, $SE_K(KP)$), along with the identity of Y and X. Z is encrypted under the public key of T, E_{TTP}. X checks all the parameters of the cheque, and if consistent, it signs and sends the cheque to Y (Step 2.3 of Figure 6). After getting the signed cheque, Y sends the key K to X as plaintext (Step 2.4 of Figure 6).

If X gets the requested good and Y gets the payment, then the protocol completes successfully, without any arbiter. However, if after getting the signed cheque in message 2.3, the provider quits the system, the downloader stands cheated. In that case, the third phase, Dispute Resolution Phase, comes into action. X sends C, Z, the cheque information signed by Y, and the signed cheque to the arbiter T. T checks the cheque, and if everything is correct, it extracts the key from Z using its private key, and sends it to X (Step 3.2 of Figure 6). At the same time, it sends the signed cheque (by X) to Y. The last step stops X from getting the key without paying.

Flaw in the Protocol:

Piva, Monteiro, and Dahab (2007) reported an attack on the File Market protocol, wherein the provider can misbehave in order to cheat the downloader. To do so, the provider encrypts the file piece with key K, (Step 2.1 of Figure 6). But in Message 2.2, instead of sending the correct key K in Z, the provider sends a garbage key K'. Following the same lines, in message 2.4, the provider sends the same garbage key K' (or any other key except the original key K) to the downloader. When the downloader gets a wrong key, it will ask the arbiter T to provide the correct key following the dispute resolution phase. The arbiter then decrypts message Z, to find the three components – identity of the provider, identity of the downloader, and the key. The arbiter is completely unaware whether the key present in the message Z is correct or not. It simply extracts the third part of the message Z (here K'), and sends it to the downloader. In such a situation, the provider gets the money, but the downloader doesn't receive the requested file. Thus the goal of the protocol, to provide fair exchange, is defeated. Piva, Monteiro, and Dahab (2007) also reports three other attacks on the File Market protocol based on timeliness, which is not in the scope of this paper.

3.6.1 Security

Forgery Prevention:

The cheque is signed by the user, thus preventing forgery of cheques (or money).

Double Spending Detection:

Accounting Centre (AC) detects double spending in the File market system. For all the registered peers, AC maintains a central account database storing information about each peer's balance, capital certificate, credit and debit amount. Periodically, peer asks AC to perform accounting operations, by sending all the cheques the peer has earned since the last accounting, to AC. AC checks the IDs and sequence numbers in the cheques to detect double-spending. If double spending is detected, the corresponding cheque is rejected, and the payer and payee are punished.

3.7 P2P-NetPay

Based on the client-server NetPay protocol (Dai and Lo, 1999), Dai and Grundy (2005) proposed P2P-NetPay micropayment protocol, which uses payword (Rivest and Shamir, 1997) chains to represent electronic coins. The protocol comprises of three phases: Registration Phase, Transaction Phase and Redemption Phase (see Figure 7). In the Registration Phase, the buyer U asks the broker B to assign payword hash chain of particular value (say *n*) to itself. B debits the buyer's account and produces the payword chain w_o, w_1, w_2, ..., w_{n-1}, w_n satisfying $w_i = h(w_{i+1})$, using a randomly-selected seed w_{n+1}. The hash function *h()* used here is MD5 (MD5 Message Digest Algorithm, 2009). The payword chain w_1, w_2, ... w_n acts as electronic coins and are given to the buyer (Step 1.2 of Figure 7), along with broker-signed touchstone *T* (consisting of coin Id Id_e and the root w_0). The seed w_{n+1} is kept as secret by the broker to prevent U from overspending and forging paywords. The broker is assumed to be honest.

The purchase of a good (of value *k*) requires the buyer to pay w_1, w_2, ..., w_k to the vendor. The buyer also sends *T*, which allows the vendor to validate the paywords using w_0, and a user-signed index *i*, representing the index of the last payword sent by U to V, which is used to prevent double spending and resolving any dispute between peers (Step 2.1 of Figure 7). The buyer U in turn gets the requested file. During the redemption phase, vendor sends all the received paywords to the broker (Step 3.1 of Figure 7), who verifies the payword using w_{n+1}, and credits vendor's account. Broker also checks for double spending. Thus, NetPay is a credit based system providing for only double spending detection.

The protocol is computationally efficient since hashing is used over encryption or signature for electronic coins. A major drawback is that it doesn't satisfy the first mandatory requirement of a P2P micropayment protocol – transferability of coins. Also, it is a fully centralized micropayment system as all the account's are debited/credited by the broker. This puts high load on the broker making it a single point of failure.

3.7.1 Security

Forgery Prevention:

The broker B keeps the seed w_{n+1} secret, thus preventing any user to forge coins.

Double Spending Detection:

As NetPay uses non-transferable coins, detecting double spending is fairly simple. Double Spending is detected by the broker B during the redemption phase. For any redemption request, firstly the broker checks the validity of the payword. If the broker generated that payword, B checks for whether the payword is earlier redeemed for a coin. Hence any kind of fraud by spender or vendor is detected, and the respective user is punished.

3.8 CPay

Jia et al. (2005) exploits the heterogeneity of the P2P network to attain load balance using transferable coins. The load of the reliable broker is reduced by introducing a new entity – Broker Assistant (BA). BA's (or eligible peers), constitutes peers having more on-line time, more bandwidth, more computational power, and more credibility. Broker selects BA's and sends them a signed certificate and a partial BA list (SetBAy) (Step 1.1 of Figure 8). An eligible peer is free to join and leave the system just like ordinary peers; but on leaving the system, broker pays them for the work done.

A peer U can buy coins from the broker B. The broker, along with coins, sends a partial list Set_U of the currently alive BA peers (Step 2.2 of Figure 8). For any transaction between buyer U and vendor X, U uses dynamic consistent hashing to map to an online BA_U from the Set_U who authorizes the transaction and detects any kind of fraud during the transaction. Dynamic consistent hashing refers to a family of hash functions, f_B: A → B, where the set of B can change dynamically. For purchasing goods, buyer U sends a request to BA_U (Step 3.1 of Figure 8), which in turn sends a signed authorization message *Au*, consisting of coin *C*, BA_U identity and BA_X identity, to vendor X (Step 3.2 of Figure 8). If peer X wants to purchase a good from peer Z, it can use the authorization message *Au* obtained from BA_U as a coin (Step 4.1 of Figure 8). Thus, in CPay, money basically can be transferred in two forms: (a) coin *C*, or (b) authorization message *Au*.

As the dynamic consistent hashing operation is expansive, it affects the system performance, while, on the other hand, dynamic consistent hashing distributes load among the several broker assistants, thus, effectively improving the system performance. A CPay broker, unlike PPay, can go offline. The CPay system is secure against double spending, as any fraud by a peer is detected by the broker assistant, and if any BA is involved in a fraud, it is detected by the broker during the redemption phase. CPay reduces the role of the broker to selling and redeeming coins, along with managing the BA's. Although the described CPay protocol doesn't preserve the anonymity of the buyer, Jia et al. (2005) also proposes two other variations of CPay – Anonymous CPay (offering anonymity to the buyer) and Group CPay (which reduces the number of the eligible peers required, by increasing each BA's work load).

3.8.1 Security

Forgery Prevention:

In CPay, money can be transferred in two forms – coin C and authorization message Au. Coin C includes the trusted broker's signature, and Au includes C. Assuming forging a signature is impossible, thus CPay prevents coin forgery.

Double Spending Detection:

The money is transferred through BA peers, thus to spend the same coin twice, a peer X needs to bribe BA_X peers such that BA_X assigns the same coin to two different peers, Y and Z. During redemption, if the broker finds that authentication message Au containing coin C is already redeemed, the double spending fraud is detected, and the culprit is punished. Moreover, the timestamp in Au, plays the same role as sequence number in PPay scheme, and helps detecting frauds where a user U goes to the broker for redemption of a coin that has been already reassigned by U.

4 Comparison Matrix of Studied Schemes

In this section, a matrix (see Table 1) is used for a comparative analysis of the P2P micropayment schemes discussed in this paper. The criteria for the comparison are the mandatory and desired requirements for a P2P micropayment scheme (as discussed in Section 2), i.e., Anonymity, Security (Double Spending Detection (DSD) or Double Spending Prevention (DSP)), Trusted Authority's Load, Transferability, and Fair Exchange.

Most of the criterion measures – *Anonymity*, *Fair Exchange*, *Complete Decentralized*, *Transferable Coins*, and *Denomination of the coins* – are binary in logic, like whether the scheme provides anonymous transaction, does the coins used in the scheme are transferable or can have different denominations, etc. The most crucial aspect of a P2P transaction is its performance, as the transactions are of very small denominations thus requiring proper resource utilization. The performance is measured in terms of the *complexity* of the system, the number of *messages* exchanged between peers and the number of expensive operations like *encryptions*, *decryptions*, *signature generation* and *signature verification* performed, for a single transaction and aggregation of payments. The other criterion of comparison is Security, which is being measured in terms of *Double Spending Detection/Prevention* and *Fair Exchange*.

This comparative analysis can help users to decide which P2P micropayment scheme to choose based on their specific requirements. For example, if a user requires a complete decentralized transaction system with double spending prevention, then it should choose Karma but the tradeoff is low-performance of the Karma scheme, whereas if a user requires Fair Exchange with good performance, then File Market is a good option. Every user has different and very specific needs and based on that, this table can help them decide which P2P micropayment scheme to implement for their system. A ranking method can be constructed by allotting points to the different measures, and thereby deciding on the best scheme for a certain user case scenario. In this paper, we have not ranked the schemes based on these criteria, as the ranking is dependent upon the system's priorities.

Also this table helps to easily visualize the void present in the currently available schemes, for example, from the table, it's certain that there is no existing scheme providing anonymity with complete decentralization. It asks the protocol designers to develop such schemes to cover all the aspects of the requirement of a potential P2P micropayment system.

5 Future Directions and Discussions

From our studies, certain observations have come to light which can help future protocol designers. The P2P micropayment protocols usually consist of three phases: Generation Phase, Exchange Phase and Redemption Phase.

Most of the schemes make use of reassignment or layering in order to transfer coins from one peer to another. However, a third strategy can be employed to get the best of both the worlds. In this scheme, the peer tries to reassign the coin through the owner. However, if it is down, then it simply layers the coin. This hybrid strategy always outperforms layered coin architecture. Also, the new strategy of reassigning coins using delegation of accountability, discussed in Catalano and Ruffo (2004), needs to be considered.

The main problem with micropayments is that the processing cost can be much higher than the transferred value. The solution lies in aggregating small payments in fewer larger payments. Moreover, any micropayment scheme must minimize the number of encryption operations needed, as it use a lot of resources. Hash chains are a better alternative. Although the decentralized schemes like Karma and Off-line Karma, are more democratic and reliable, it is infeasible to implement them because of the huge bandwidth they will consume. For commercial application of a P2P micropayment scheme, it is necessary to provide for transferable coins and scalability, with minimum load on the broker (the trusted authority).

6 Conclusion

We have presented a comprehensive survey of Peer2Peer Micropayment schemes, while at the same time laying down the requirements that need to be kept in mind while designing such schemes. These requirements (both mandatory and desired) might help protocol designers in coming up with new schemes in the future. We have also presented a matrix that can be used for comparing the different schemes. The future directions might help in coming up with schemes which are not only highly efficient, but also provide utmost security. The failure of micropayment makes the case for P2P micropayment stronger and challenging. We hope that this work will help researchers who are new to this field. Our work tries to accumulate the existing research and gives a way how the P2P micropayment can be implemented in a real world scenario. To the best of our knowledge, no such survey and analysis of Peer2Peer Micropayments is available in literature.

References

Warner Bros (2008). Warner Bros. Obtained through the Internet: http://www.msnbc.msn.com/id/12694081, [accessed on 3/6/2008].

Catalano, D. and Ruffo, G. (2004) 'A Fair Micro-Payment Scheme for Profit Sharing in P2P Networks', *International Workshop on Hot Topics in Peer-to-Peer Systems*, pp. 32–39, IEEE Computer Society.

Chaum, D. and Heyst, E. (1991) 'Group Signatures', *Advances in Cryptology - EUROCRYPT*, pp. 257–265, Springer.

Chaum, D. and Pedersen, P. (1992) ‘Transferred Cash Grows in Size’, *EUROCRYPT*, pp. 390-407, Springer.

Ruffo, G., Catalano, D. and Schifanella, R. (2005) ‘A P2P Market Place Based on Aggregate Signatures’, *International Workshop on Applications and Economics of Peer-to-Peer Systems*, pp. 54–63, Springer.

Garcia, F. and Hoepman, J. (2005) ‘Off-Line Karma: A Decentralized Currency for static Peer-to-Peer and Grid Networks’, *International Conference on Applied Cryptography and Network Security*, pp. 364–377, Springer.

Jia, Z., Tiange, S., Liansheng, H. and Yiqi, D. (2005) ‘A New Micro-payment Protocol Based on P2P Networks’, *IEEE International Conference on e-Business Engineering*, pp. 449–455, IEEE Computer Society.

Wei, K., Chen, Y., Smith, A. and Vo, B. (2006) ’WhoPay: A Scalable and Anonymous Payment System for Peer-to-Peer Environments’, *IEEE International Conference on Distributed Computing Systems*, IEEE Computer Society.

Kou, W. (2003) ‘Micropayments - Payment Technologies for E-Commerce*’*, pp. 245–280, Springer.

Micali, S. (2003) ‘Simple and Fast Optimistic Protocols for Fair Electronic Exchange’, *Annual Symposium on Principles of Distributed Computing*, pp. 12–19, ACM.

Wikipedia (2008). Micropayments. Obtained though the Internet:
http://en.wikipedia.org/wiki/Micropayments,
[accessed on 31/5/2008].

EMI Music (2008). EMI Music. Obtained through the Internet:
http://www.emigroup.com/Press/2006/press25.htm,
[accessed on 3/6/2008].

Odlyzko, A. (2003) ‘The Case against Micropayments’, *International Conference on Financial Cryptography*, pp. 77–83, Springer.

Ruffo, G. and Schifanella, R. (2005) ‘Scalability Evaluation of a Peer-to-Peer Market Place based on Micro Payments’, *International Workshop on Hot Topics in Peer-to-Peer Systems*, pp. 10–17, IEEE Computer Society.

Someren, N., Odlyzko, A., Rivest, R., Jones, T. and Goldie-Scot, D. (2003) ‘Does Anyone Really Need MicroPayments?’, *International Conference on Financial Cryptography*, pp. 69–76, Springer.

Vishnumurthy, V., Chandrakumar, S. and Sirer, E. (2003) ‘KARMA: A Secure Economic Framework for Peer-to-Peer Resource Sharing’.

Yang, B. and Garcia-Molina, H. (2003) ‘PPay: Micropayments for Peer-to-Peer Systems’, *Conference on Computer and Communications Security*, pp. 300–310, ACM.

Zuo, M. and Li, J. (2005) ‘Constructing Fair-Exchange P2P File Market’, *International Conference on Grid and Cooperative Computing*, pp. 941–946, Springer.

Dai, X. and Grundy, J. (2005) ‘Off-line Micro-payment System for Content Sharing in P2P Networks’, *International Conference on Distributed Computing and Internet Technology*, pp. 297–307, Springer.

Rivest, R. and Shamir, A. (1997) ‘PayWord and MicroMint: Two Simple Micropayment Schemes’. *1996 International Workshop on Security Protocols*, Lecture Notes in Computer Science, Vol. 1189, pp. 69–87, Springer.

Ruffo, G. and Schifanella, R. (2007) ‘FairPeers: Efficient Profit Sharing in Fair Peer-to-Peer Market Places’, *Journal of Network and Systems Management*, Volume 15, Number 3, pp. 355-382.

MD5 (2008). The MD5 Message-Digest Algorithm. Obtained through the Internet: *http://tools.ietf.org/html/rfc1321* , [accessed on 12/3/2008].

Dai, X. and Lo, B. (1999) ‘NetPay – An Efficient Protocol for Micropayments’, *WWW Fifth Australian World Wide Web Conference*, Australia.

Glassman, S., Manasse, M., Abadi, M., Gauthier, P. and Sobalvarro, P. The Millicent Protocol for Inexpensive Electronic Commerce. Obtained through the Internet:
http://www.w3.org/Conferences/WWW4/Papers/246,
[accessed on 8/8/2008].

Geer, D. (2004) ‘E-micropayments sweat the small stuff’, *Computer*, 37(8), August 2004.

Piva, F., Monteiro, J. and Dahab, R. (2007) ‘Strand spaces and fair exchange: More on how to trace attacks and security problems’, *SBSeg2007: VII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*.

Hoepman, J. (2006) ‘Distributed Double Spending Prevention’, *CoRR Journal*, Volume abs/0802.0832.

Lesk, M. (2004) ‘Micropayment: An Idea Whose Time has Passed Twice?’, *IEEE Security and Privacy*, Volume 2, Issue 1, pp. 61-63.

Napster (2008). The Napster Homepage. Obtained through the Internet:
http://www.napster.com, [accessed on 22/10/2008].

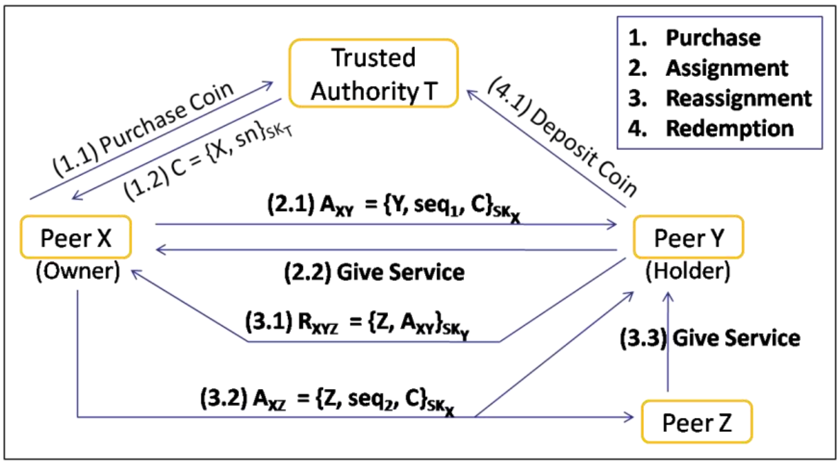

Figure 1: PPay Basic Micropayment Scheme

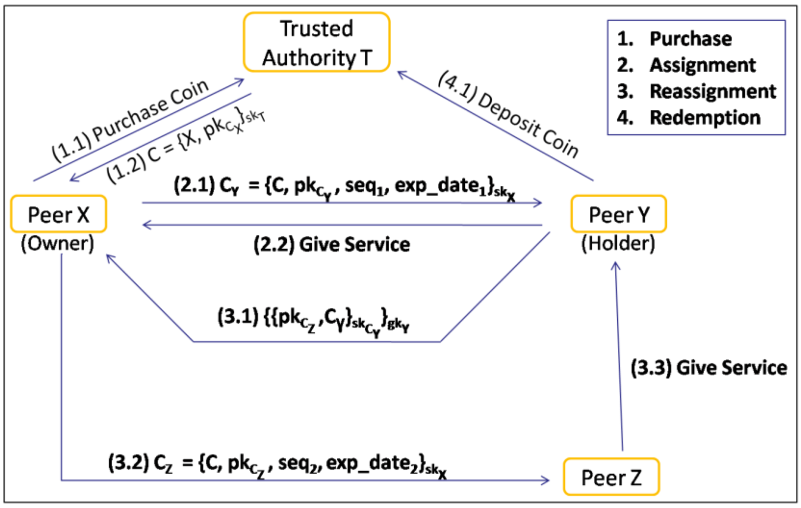
Trusted Authority T
1. Purchase
2. Assignment
3. Reassignment
4. Redemption
(1.1) Purchase Coin
(1.2) $C = \{X, pk_{C_X}\}_{sk_T}$
(4.1) Deposit Coin
(2.1) $C_Y = \{C, pk_{C_Y}, seq_1, exp\_date_1\}_{sk_X}$
Peer X
(Owner)
Peer Y
(Holder)
(2.2) Give Service
(3.1) $\{\{pk_{C_Z}, C_Y\}_{sk_{C_Y}}\}_{gk_Y}$
(3.3) Give Service
(3.2) $C_Z = \{C, pk_{C_Z}, seq_2, exp\_date_2\}_{sk_X}$
Peer Z

Figure 2: WhoPay Basic Micropayment Scheme

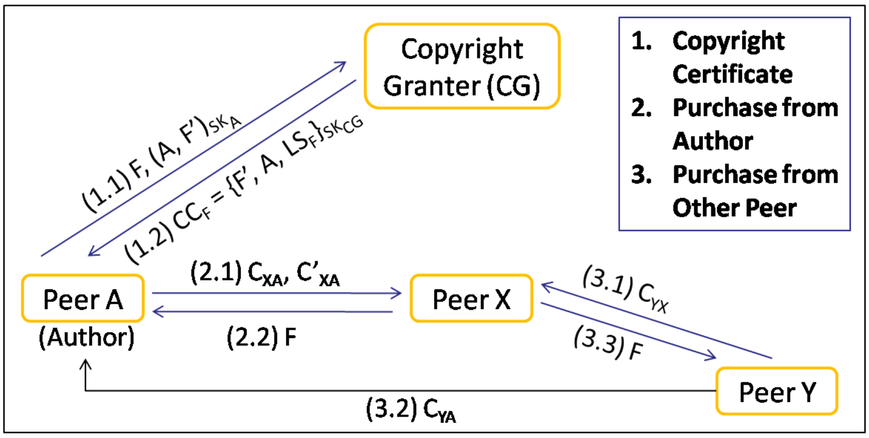

Figure 3: FairPeers Micropayment Scheme

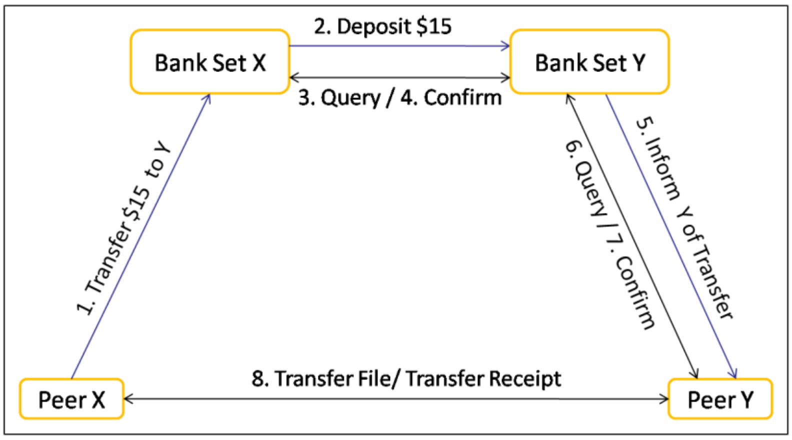

Figure 4: Karma Micropayment Scheme

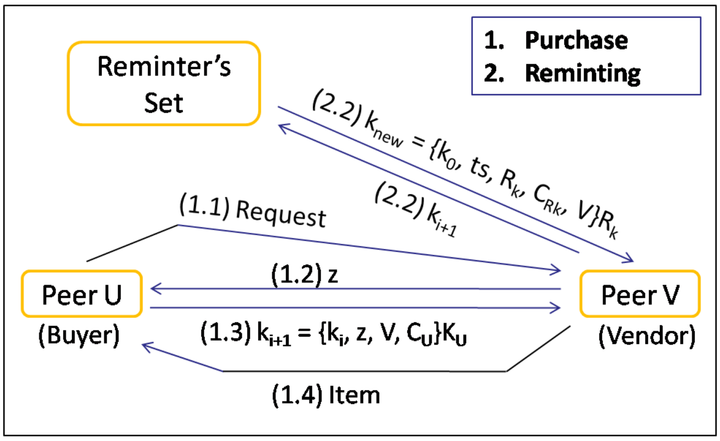

Figure 5: Off-line Karma Micropayment Scheme

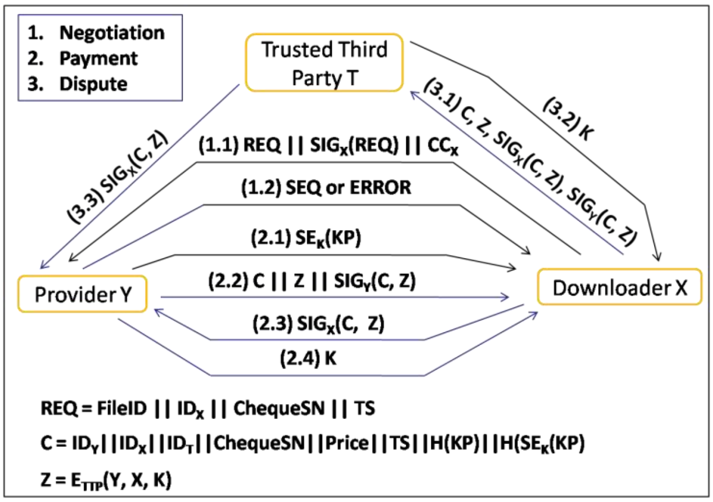
1. Negotiation
2. Payment
3. Dispute
Trusted Third Party T
(3.3) $SIG_X(C, Z)$
(3.1) C, Z, $SIG_X(C, Z)$, $SIG_Y(C, Z)$
(3.2) K
(1.1) REQ || SIG_X(REQ) || CC_X
(1.2) SEQ or ERROR
(2.1) $SE_K(KP)$
(2.2) C || Z || $SIG_Y(C, Z)$
(2.3) $SIG_X(C, Z)$
(2.4) K
Provider Y
Downloader X
REQ = FileID || ID_X || ChequeSN || TS
C = ID_Y||ID_X||ID_T||ChequeSN||Price||TS||H(KP)||H(SE_K(KP))
Z = E_{TTP}(Y, X, K)

Figure 6: Fair Exchange File Market Micropayment Scheme

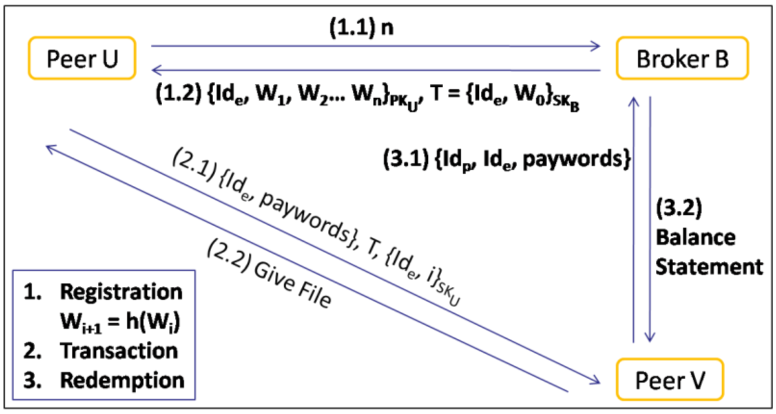
Peer U
Broker B
Peer V
(1.1) n
(1.2) $\{Id_e, W_1, W_2 \ldots W_n\}_{PK_U}$, $T = \{Id_e, W_0\}_{SK_B}$
(2.1) $\{Id_e$, paywords$\}$, T, $\{Id_e, i\}_{SK_U}$
(2.2) Give File
(3.1) $\{Id_p, Id_e$, paywords$\}$
(3.2) Balance Statement
1. Registration
$W_{i+1} = h(W_i)$
2. Transaction
3. Redemption

Figure 7: P2P-NetPay Micropayment Scheme

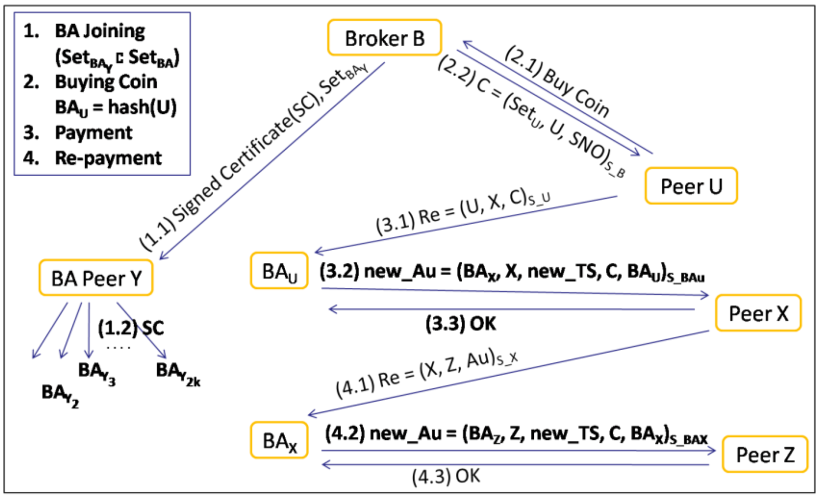
1. BA Joining
$(Set_{BA_Y} \subset Set_{BA})$
2. Buying Coin
BA_U = hash(U)
3. Payment
4. Re-payment
Broker B
(1.1) Signed Certificate(SC), Set_{BA_Y}
(2.1) Buy Coin
(2.2) $C = (Set_U, U, SNO)_{S\_B}$
Peer U
BA Peer Y
(1.2) SC
BA_{Y_2}
BA_{Y_3}
....
$BA_{Y_{2k}}$
(3.1) $Re = (U, X, C)_{S\_U}$
BA_U
(3.2) $new\_Au = (BA_X, X, new\_TS, C, BA_U)_{S\_BAu}$
(3.3) OK
Peer X
(4.1) $Re = (X, Z, Au)_{S\_X}$
BA_X
(4.2) $new\_Au = (BA_Z, Z, new\_TS, C, BA_X)_{S\_BAX}$
(4.3) OK
Peer Z

Figure 8: CPay Micropayment Scheme

Criteria	PPay	WhoPay	FairPeers	Karma*	Off-line Karma	File Market	P2P-NetPay	CPay
Based on	-	PPay	PPay	-	Karma	-	PayWord	-
Anonymity	No	Yes	No	No	No	No	No	Yes
Decentralized	No	No	No	Yes	Yes	No	No	No
Transferable Coins	Yes	Yes	Yes	Yes	Yes	No	No	Yes
Coin of Any Denomination	No	No	No	Yes	No	Yes	No	No
Fair Exchange	No	No	No	Yes	No	Yes	No	No
Security**	DSD	DSD	DSD	DSP	DSD	DSD	DSD	DSD
Complexity*	O(c)	O(c)	O(c)	$O(p^2)$	O(p*r)	O(p)	O(t)	O(b)
Messages$^+$	3	2	6	$O(n^2)$	2	4	1	2
Encryption$^+$	-	-	-	-	-	2	-	-
Decryption$^+$	-	-	-	-	-	1	-	-
Signature Generation$^+$	2	3	4	$O(n^2)$	1	2	1	2
Signature Verification$^+$	4	4	8	$O(n^2)$	2	2	2	5

Table 1: The comparison matrix of P2P micropayment schemes

**c = number of coins, t = number of transactions, p = number of peers, r = size of remitter set, b = number of broker assistant, n = size of the bank set*

***DSD = Double Spending Detection, DSP = Double Spending Prevention*

$^+$Number of Messages, Encryption, Decryption, Signature Generation and Verification are for a single transaction (excluding the messages for request and delivery of goods)