Peer2Peer (P2P) Micropayments: A Survey and Critical Analysis

Mohit Jain* and Siddhartha Lal*

Faculty Advisor: Prof. Anish Mathuria*

Dhirubhai Ambani Institute of Information and Communication Technology
Gandhinagar, Gujarat 382007
Email: [miohit_jain, siddhartha_lal, anish_mathuria]@daiict.ac.in

***Abstract*— Rapidly emerging incentive driven P2P networks, along with several commercial P2P applications, necessitate secure and efficient payment methods which maintain fairness in a profit sharing environment and reward peers who share their resources. This has given a fresh lease of life to micropayments schemes, which are now being used to drive such networks. These schemes are unique in the sense that they have to cater to several requirements which are peculiar to the P2P scenario. 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 requisites that need to be kept in mind while designing such schemes, present a survey of existing schemes, and analyze them critically. We also model an existing scheme [Zuo 05] using formal analysis tools (CSP and FDR) and prove the existence of a flaw in it. The scheme is then fixed and proven to be secure using the same formal analysis.**

I. INTRODUCTION

The availability of low cost digital content on the internet triggered the advent of micropayment schemes during the mid nineties of the 20th century. These schemes were the outcome of a change in the mindset of Internet content providers. Earlier, 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, opportunities increased exponentially and content providers began to seek profit for their services. All of them wanted a share of the pie known as the Dot Com Boom. 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 [Micropayments]. Also, these payment methodologies involved substantial delay, user involvement, and potential for disputes [Kou 03]. This led to substantial research in the field of Micropayments and several schemes were proposed for implementing them.

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 25 cents. Hence going by this definition, micropayments are payments of value less than or equal to 20 cents. With increased interest in selling intellectual property over the World Wide Web, micropayments have become an important area of research. 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. Utmost security is not an essential requirement and fraudulent users are deterred by making cheating detectable, traceable, and unprofitable. Therefore, micropayments mostly seek to provide optimum security while maintaining high efficiency.

However, in recent times, several arguments have been made against micropayment methodologies, notably [van Someren 03]. Most of these arguments have had a sociological, economical, and psychological basis. [Odlyzko 03] 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 materialized and quite a few new ones keep sprouting up. However, it is circumspect whether venture capitalists will continue investing in a prospect that doesn't seem to guarantee *immediate* returns.

Very recently, however, micropayments have gained a fresh perspective in the Peer2Peer (P2P) scenario. Several commercial P2P applications are being launched and there are talks of big corporate houses collaborating with existing networks [Bros]. 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 03][1]. There are several strong reasons for doing so.

[1]PPay is short for PeerPay

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 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 04]. Thirdly, up till now, P2P networks have received a lot of flak from commercial entities because of copyright issues. By incorporating micropayments with P2P technologies, we can overcome this problem and turn P2P into a commercial platform [Bros], [Music]. Fourthly, P2P networks are usually self-organized and robust, mostly independent of centralized servers, scalable, and opulent in resources at the edge of the network [Zuo 05]. These are some of their most inviting features and can help to make micropayments scalable and economically feasible in real world applications.

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 issues. While analyzing one of the schemes [Zuo 05], we came across certain defects. We use formal analysis tools (CSP and FDR) to prove the existence of the aforementioned flaws (Section 5). Methods to remedy those flaws are suggested, along with a formal analysis of the fixed protocol. To the best of our knowledge, no such research work on P2P Micropayments exists and it is hoped that this study will benefit and assist researchers who are new to the field.

II. 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.

A. Mandatory Requirements

1) Transferability: Since in a P2P network, peers can be consumers as well as vendors, it is essential that the payment mechanism (electronic coin, electronic lottery ticket, electronic cheque) be transferable 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, while also increasing the size of the coins [Chaum 92]. However, [Yang 03] makes use of peers in order to implement transferable coins that do not increase in size.

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 fool proof 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)* in the number of transactions. Offline payments are preferred because they have *lower latency, communication costs and computational costs* [Yang 03].

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 03], 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.

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. In conventional 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*.

B. Desired Requirements

1) Anonymity: One of the most striking features of micropayments was their ability to ensure anonymity of online monetary transactions. However, experience with schemes providing anonymity [Kai Wei 06], [Jia 05] 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 03]. 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) Double Spending Prevention: Double spending prevention, as the term implies, refers to preventing coin fraud at its outset. 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 03] 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.

3) 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 04]. This requirement is different than Fair-Exchange (discussed in the next sub section) 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 (which had so far kept P2P networks out of the legal business).

4) 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 them from doing so, 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 03] and some of the schemes [Zuo 05] that we studied make use of them in order to ensure fair exchange of goods in a P2P market.

III. Current 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 fetures.

A. PPay

PPay [Yang 03] is one of the pioneering works in the field of P2P micropayments and also forms the basis of several other schemes such as [Kai Wei 06] and [Catalano 04]. It makes use of self managed and floating coins in order to minimize the involvement of a trusted authority (T for convenience). The coins can move or float from one peer to another and all the security issues are handled by the owner of the coin. T is involved only when the coin is created or cashed and hence its load is *O (n)* in the number of floating coins. Double spending is possible but it is made unprofitable by ensuring traceability of the malicious peer. We present a basic version of the PPay protocol here. For a detailed discussion of the security issues or extensions to the scheme, please refer to [Yang 03].

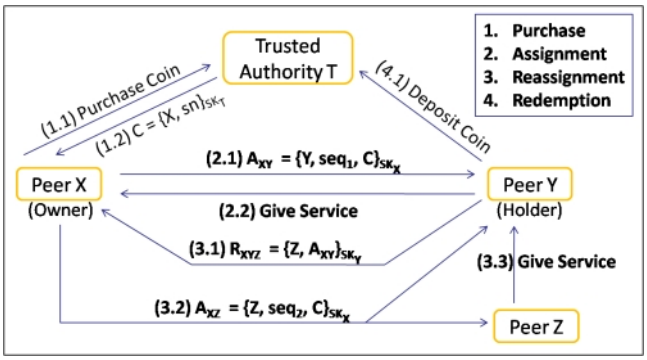

Fig. 1. PPay Basic Protocol

In the basic PPay protocol, each user (say X) buys a certain number of coins from the Trusted Authority (here T). These coins are messages of the form C = X, sn which are signed by T for authenticity. The serial number sn is unique for each coin. In order to pay for a service, X can give this coin to the vendor Y by signing the message Y, seq1, C with its private key. 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 (as shown in Step 3.1 of Figure 1). The owner X now reassigns the coin to Z by signing the message Z, seq2, C. seq2 is greater than seq1 in order to prevent double spending. The previous assignment 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.

B. WhoPay

[Kai Wei 06] can be understood as PPay with anonymity. WhoPay uses Group Signature [Chaum 91] to provide anonymity and to reveal the identity of a node in case a fraud is detected. A malicious node's identity can be revealed by the joint effort of Judge (who holds the master private key) and Trusted Authority T. Any transaction in WhoPay requires the vendor (say, V) to generate a new public-private key pair (pkCV, skCV). Spender signs the coin C, along with pkCV , to assign the coin to V. Thus, instead of representing coins by a serial number (as in PPay), they are represented by a public key and the peer, who knows the corresponding private key, is the holder of the coin. In WhoPay, any message from the coin owner (who purchased the coin from the Trusted Authority) is signed by the peer's private key, whereas, any message from the coin holder (who obtained the coin from another peer) is signed using two keys, the coin private key, which proves the holdership of the coin and the peer's group private key.

C. Aggregate Signature

As per [Ruffo 05], allowing users to mint coins in a layered coin architecture, with number of allowed layers ranging from six to eight, is the best possible option for a P2P Micropayment scheme. To overcome the demerits of layered coins in the PPay model, [Dario Catalano 05a] proposes to use Aggregate Signature (AS) [Dario Catalano 05b], instead of the RSA signatures used in PPay Layered Coins model. AS signatures are advantageous over RSA as the growth of coin size is minimal after each reassignment. Moreover, the computational cost of signature verification at the broker's end is considerably lesser. [Dario Catalano 05a] concludes that for a fraud rate less than 5%, the best strategy is to use Layered coins with aggregate signatures, while allowing users to mint coins (after getting a certificate from the Trusted Authority). If the fraud rate becomes greater than 5%, the performance of aggregate signatures is worse than RSA and the broker's load is increased considerably.

D. Karma

To the best of our knowledge, [Vishnumurthy 03] is the first protocol to offer a completely decentralized P2P micropayment scheme, and lays the foundation of protocols like [Garcia 05]. The money held by each peers is symbolized by its Karma. Each peer (say, A) is associated with a bank-set (bankA), calculated using a hash function, which consists of a set of nodes (peers) that keeps track of A's Karma. The bank-set represents the semi Trusted Authority in this case. Any transaction between the peers involves the peers' bank-set interacting with each other to transfer Karma from one peer to another. As there is no Trusted Authority, each decision is taken by a majority voting. This leads to a lot of messages and computational overhead. However, this overhead utilizes the resources of the peers and thus, supposedly, maintains efficiency.

E. Fair Exchange File Market

This scheme [Zuo 05] borrows ideas from [Micali 03] 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). 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 auditing 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.

IV. Comparison Matrix of Studied Schemes

In this section we use a matrix to compare the P2P micropayment schemes that we have studied. The criterions for the comparison are the requirements for such schemes, i.e., Anonymity, Security (Double Spending Detection or Prevention), Trusted Authority's Load, Transferability, and Fair Exchange. Apart from these, we have also taken into account certain performance issues like number of signature operations (generation and verification), number of messages exchanged between peers for each transaction, and aggregation of payments.

TABLE I
Survey of Studied Schemes

	PPay	WhoPay	AggSign	Karma[a]	FairEx
Messages	3	2	1	$O(k^2)$	7
Signature Gen.	2	3	1	1, O(k)	3
Signature Ver.	4	4	2	$O(k/k^2)$	4
Central Control	Y	Y	Y	N	Y
Fair Exchange	N	N	N	N	Y
Anonymity	N	Y	N	N	N
Security	D	D	D	P	D
Single/Multiple	S	S	S	M	M
Based On	-	PPay	PPay	-	-
Performance[b]	O(c)	O(c)	O(t)	$O(n^2)$	O(n)

[a] k = Size of Bank Set

[b] c = No. of coins, n = No. of nodes, r = Size of Remitter Set, t = No. of transactions

V. Breaking Fair Exchange File Market

While analyzing one of the schemes, [Zuo 05], we came across certain defects. Using formal analysis tools (CSP and FDR), we prove the existence of the aforementioned flaw in the scheme. Since the scheme guarantees fair exchange in a P2P file market, we studied fair exchange protocols in order to remedy it. After fixing the defect, we used the same formal analysis (CSP and FDR) to prove that the new scheme was secure against such attacks. The following subsections talk about the flaw and the fix in greater detail.

A. Fair Exchange

The basic requirement of a fair exchange protocol is that under all circumstances, either the two parties, i.e. the vendor and the customer, get what they expect or none does. One can achieve such a fair exchange by allowing every transaction to be authorized by a trusted third party (TTP). The major drawback of such a solution is that the TTP needs to be online at all times, even when the two parties are honest. For a P2P system, the TTP would act as a performance bottleneck. Such fair exchange protocols are termed as Non-Optimistic Fair Exchange Protocol.

It would be better to involve the TTP only when a dispute arises between the customer and the vendor. Such fair exchange protocols are termed as Optimistic Fair Exchange Protocols. They are optimistic about the fact that most of the transactions occur between honest parties. Hence the involvement of an arbiter in each transaction isn't necessary. The optimistic fair exchange can seek to provide two kinds of fairness: strong fairness or weak fairness [Asokan 97]. An optimistic protocol achieves strong fairness if it provides either strong revocability or strong generatability.

As per [Asokan 97], revocability refers to the ability of the TTP to revoke the payment made by a customer in case it does not receive the goods. On the other hand, generatability refers to the ability of the TTP to regenerate the item without the cooperation of the vendor (in case the customer is not satisfied with the merchandise).

One can also classify a fair exchange protocol in terms of the features it provides, based on the Atomicity Properties: Money Atomicity, Goods Atomicity, and Certified Delivery. Money atomicity ensures that each credit operation is matched with an equal amount of debit operation. Goods atomicity [Adi 00] in an electronic commerce protocol rules out the following two cases: a) a customer pays but doesn't get the items, b) customer gets the items without paying the vendor. Meanwhile, certified delivery assures that the customer and the vendor can prove the contents of what was delivered. The atomicity requirements form a hierarchy. Each step must be completed to proceed to the next one. Money atomicity is the weakest property, as even the delivery of items is not ensured. Goods atomicity automatically provides money atomicity but doesn't talk about correctness of the exchanged items. Certified delivery provides goods atomicity, taking the quality of the content into consideration as well. Hence, one can say that a fair exchange protocol must seek to provide certified delivery. For a brief overview of fair exchange protocols and the properties they need to adhere to, we suggest [Gehlot 07] as a reference.

B. Fair Exchange File Market

1) Goals: File Market Protocol claims to achieve optimistic fair exchange between two parties: namely, the provider (P) and the downloader (D). As explained in the previous subsection, in optimistic fair exchange, a Trusted Third Party (T) is involved only in case of a dispute, as an arbiter. The File Market Protocol tries to achieve Goods Atomicity and Certified Delivery.

2) Description: The Fair Exchange File Market Protocol basically consists of 3 phases. The third phase occurs only in case of disputes. The first step is the Negotiation Phase, in which the downloader D sends the download request to the provider P (see Figure 2, Messages labeled 1.1 and 1.2). The download request consists of the id of the file which D wants to download, D's own identity, serial number of the cheque (which D promises to give to P after receiving the requested goods) and the timestamp. To prevent repudiation, D signs all these and sends to P, along with its own Capital Certificate CC (to prove that it has enough money to pay for the goods).

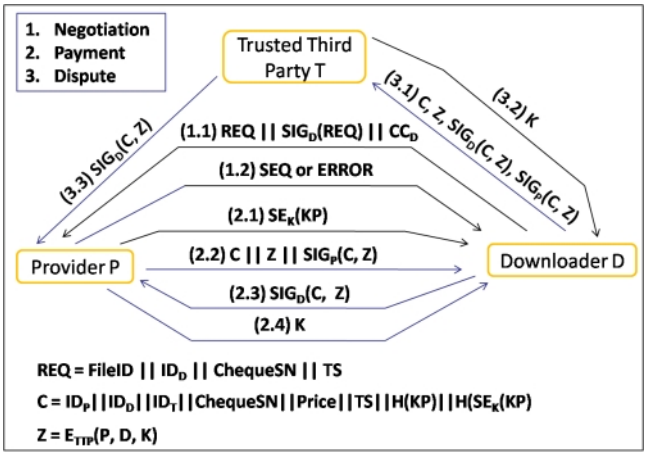

Fig. 2. Fair Exchange File Market - Original Protocol

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 becomes a part of the network. CC is a digital certificate that states the maximum amount the downloader can pay using cheques. After receiving Message 1.1 (refer Figure 2), P

verifies the signature on the download request and the capital certificate. If P is satisfied, it encrypts one of the pieces of the file, and sends its sequence number to D. Else it sends an error message (Message 1.2).

The second phase is the Payment Phase. After getting all the pieces of the file (including the encrypted file piece, SEK(KP)), P signs the cheque and sends it to be signed by D (Message 2.2). The cheque information consists of two parts: C and Z. The first part C, contains the identity of the provider P, the downloader D, and the trusted third party (T), along with serial number of cheque, the amount to be paid, the hash of the key piece, and the hash of the encrypted key piece. The second part Z, consists of the key K (used to decrypt the encrypted file piece, SEK(KP)), along with the identity of P and D. Z is encrypted under the public key of T. D checks all the parameters of the cheque. If they are consistent, it signs and send the cheque to P (Message 2.3). After getting the signed cheque, P sends the key K to D as plaintext.

If D gets the key and P gets the amount, 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. D sends C, Z, the cheque information signed by P, 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 D (Message 3.2). At the same time, it sends the signed cheque (by D) to P. The last step stops D from getting the key without paying.

C. Flaw in the Protocol

In the File Market Protocol, the provider can misbehave in order to cheat the downloader. To do so, the provider encrypts the file piece with key K, in message 2.1 (refer Fig. 2). 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 (i.e. via the dispute resolution phase). The arbiter then decrypts the message Z, to find the three components, namely: identity of the provider, identity the of 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 message Z, i.e. K', and sends it to the downloader. In such a situation, the provider gets the money, but the downloader doesn't receive the expected file. Thus the goal of the protocol, to provide fair exchange, is defeated.

D. Formal Analysis using CSP-FDR

For the formal analysis of the File Market protocol, we used CSP and FDR [Wehrheim 02]. FDR (Failures-Divergence Refinement) is a model checking tool based on CSP (Communicating Sequence Process) theory of concurrency. The protocol, to be checked, is modeled using CSP, and fed into FDR, which produces a counter example if any one of the specification isn't met. In CSP, each entity such as downloader, provider, TTP, etc. is represented as a process. A process performs an event, or a sequence of events, to move from one state to another. Any two processes communicate using a communication channel. The channels are named as x(in/out)y, where in/out refers to the direction (relative to x), and y is the other party involved in the communication.

Let us look at the File Market protocol, as modeled in CSP. In the first message (Message 1.1) of the protocol, the downloader sends a download request to the provider. If the provider accepts the request, it sends the encrypted goods to the downloader (Message 2.1), else it doesn't send anything. The provider then sends the cheque details to the downloader (Message 2.2). The downloader signs the cheque and sends the signed cheque back to the provider (Message 2.3), thereby confirming the payment. After validating the cheque, the downloader sends the key to the downloader (Message 2.4) as plaintext. In case of any dispute, the downloader sends the transaction details to the TTP (message 3.1). Subsequently, TTP sends the key to the downloader (Message 3.2) and the cheque to the provider (Message 3.3).

The three processes are modeled in CSP as follows:

1) The Downloader Process:

```
DOWNLOADER = ABORT |~| (doutp !downloadReq
-> DOWNLOAD_REQ_SENT)

DOWNLOAD_REQ_SENT = ABORT |~|
(dinp ?x -> (if x==encryptedGoods
then WAIT_CHEQUE_DETAILS else
DOWNLOAD_REQ_SENT))

WAIT_CHEQUE_DETAILS = ABORT |~| (dinp ?x
-> (if x==chequeDetails then SEND_CHEQUE
else WAIT_CHEQUE_DETAILS))

SEND_CHEQUE = ABORT |~| (doutp !cheque
-> CHEQUE_SENT)

CHEQUE_SENT = (dinp ?y -> (if
(y==key) then CHECK_KEY_P else
CHEQUE_SENT)) [](timeoutEvent -> doutt
!transactionDetails -> TTP_QUERIED)
```

```
TTP_QUERIED = dint ?y -> (if (y==key) then
CHECK_KEY_T else TTP_QUERIED)
CHECK_KEY_P = CORRECT_KEY |~|
INCORRECT_KEY_P

CORRECT_KEY = correctKey -> SUCCESS

INCORRECT_KEY_P = doutt
!transactionDetails -> TTP_QUERIED

CHECK_KEY_T = CORRECT_KEY |~|
INCORRECT_KEY_T

INCORRECT_KEY_T = FAIL
```

2) The Provider Process:

```
PROVIDER = ABORT |~| (pind ?x -> (if
x==downloadReq then DOWNLOAD_REQ_REC else
PROVIDER))

DOWNLOAD_REQ_REC = ABORT |~| (poutd
!encryptedGoods -> ENCRYPTED_GOODS_SENT)

ENCRYPTED_GOODS_SENT = ABORT |~| (poutd
!chequeDetails -> CHEQUE_DETAILS_SENT)

CHEQUE_DETAILS_SENT = ABORT |~| (pind
?x -> (if x==cheque then SEND_KEY else
CHEQUE_DETAILS_SENT))

SEND_KEY = ABORT |~| (poutd !key -> STOP)

GET_CHEQUE_T = FAIL |~| (pint ?x -> (if
(x==cheque) then (END) else GET_CHEQUE_T))
```

3) The TTP Process:

```
TTP = WAIT_TRANS_DETAILS

WAIT_TRANS_DETAILS = tind ?x -> (if
x==transactionDetails then OK_TRANSACTION
else WAIT_TRANS_DETAILS)

OK_TRANSACTION = toutd !key ->
SEND_CHEQUE_D

SEND_CHEQUE_D = toutp !cheque -> STOP
```

VI. Fixing the Protocol

To fix the aforementioned flaw, a fourth entity, namely Judge, is introduced. The judge differs from the trusted third party. The trusted third party need not be a central entity. It can be any of the peers which is trusted by both downloader and provider. But the judge needs to be an honest central authority, having special powers to shun a cheating peer from the P2P network. After a complete run of the protocol, if the downloader gets a wrong key, it can move to the fourth phase, the Judgment Phase. In this phase, the downloader sends the complaint (namely COMP) to the judge:

```
COMP = C, Z, SIGD(C, Z), SIGP(C, Z), REQ,
SEK(KP)
```

The judge validates the cheque, the request, and the encrypted key piece by using the key sent. If the judge finds that the provider misbehaved by providing the wrong key, then the fraudulent provider is punished.

The above solution still suffers from a flaw. Here, the downloader can misbehave in order to gain advantage. The downloader can send a wrong request REQ' to the judge. Then, even if the judge decrypts the encrypted key piece SEK(KP), using the correct key K, it will find that the request and the file don't match. Hence, the judge will find the provider to be fraudulent and punish it.

The solution to this flaw is that at some point in the protocol, the provider must sign what it's promising to send the downloader. Thus, in order to get both the parties sign the request, we include the REQ in the message C. We also observed that the hash of the key piece H(KP) is not being used anywhere in the protocol. Hence it can be removed from the message C. The hash of encrypted file piece H(SEK(KP) is used in to check the goods received in Message 2.1. Thus the new message, denoted as C*, is as followws:

```
C* = IDP||IDD||IDT||ChequeSN||Price||TS||
H(SEK(KP) || REQ
```

Now the downloader can't cheat the provider, thereby providing an optimistic fair exchange.

A. Formal Analysis of Fix

The judge comes into action only when a downloader registers a complaint against a provider. This is modeled in CSP as follows:

```
  CHECK_KEY_T = CORRECT_KEY |~|
INCORRECT_KEY_T

INCORRECT_KEY_T = douta !complain ->
COMPLAIN_SENT

COMPLAIN_SENT = SUCCESS
```

If the judge receives a valid complain from the downloader, it can punishe the cheating peer by banning it from the network.

1) The Judge Process:

```
JUDGE = WAIT_FOR_COMPLAIN

WAIT_FOR_COMPLAIN = jind ?x -> (if
x==complain then DO_JUDGEMENT else
WAIT_FOR_COMPLAIN)

DO_JUDGEMENT = providerBan -> END |~|
invalidComplain -> END
```

The modified File Market protocol provides both money atomicity and certified delivery. The specification is as follows:

```
  SPEC1 = (STOP |~| ((dinp.encryptedGoods
-> STOP)
(dinp.encryptedGoods -> dinp.key ->
correctKey -> STOP)
(dinp.encryptedGoods -> dint.key ->
correctKey -> STOP)
(dinp.encryptedGoods -> dint.key ->
providerBan -> STOP)
(dinp.encryptedGoods -> dint.key ->
invalidComplain -> STOP)
(dinp.encryptedGoods -> dinp.key ->
dint.key -> correctKey -> STOP)
(dinp.encryptedGoods -> dinp.key ->
dint.key -> providerBan -> STOP)
(dinp.encryptedGoods -> dinp.key ->
dint.key -> invalidComplain -> STOP)))


  SPEC2 = (STOP |~| ((aind.complain ->
providerBan -> STOP)
(aind.complain -> invalidComplain ->
STOP)))
```

The trace of the following specification in FDR proves that the fixed protocol is free from the discussed attack.

B. Security Analysis

The enhanced file market protocol provides strong fairness to the provider, but weak fairness to downloader. As mentioned in [Asokan 97], an optimistic fair exchange protocol, at its best, can only provide weak fairness to the initiator (here the downloader) of the exchange. However it can ensure strong fairness for the other party involved, the provider in our case. The provider achieves strong fairness, because it sends the key only after getting the cheque. Whereas in case of the downloader, if it gets a wrong key, revocability or generatability by trusted third party T is not provided for. The downloader can only lodge a complaint with the judge. If the judge finds the complaint valid, proper action could be taken against the cheating peer (like shunning it from the network). Since the protocol is for micropayments, the downloader suffers a small loss. On the other hand, the provider is banned from the network for a stipulated period of time. Thus the risks involved in a fraud far outweigh the benefits.

VII. 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 a new scheme. We have also used formal analysis in order to discover a flaw in one of the protocols [Zuo 05] and then used the same analysis to show that the fixed protocol is secure against such attacks. This analysis can be used to find out areas where P2P micropayments might be defective. We have also presented a matrix that can be used for comparing the different schemes. This comparison matrix can be used to determine which scheme best suits one's requirements. We hope that this work will help researchers who are new to this field. To the best of our knowledge, no such survey and analysis of Peer2Peer Micropayments is available in literature.

VIII. Acknowledgements

We are sincerely indebted to our faculty advisor Dr. Anish Mathuria for providing us with valuable insights during the course of this research work. Without his impeccable guidance and suggestions, this work would be far from completion. We are also thankful to Mr. Jai Gehlot for helping us out with the formal analysis of the protocols.

References

[Adi 00] Kamel Adi, Mourad Debbabi & Mohamed Mejri. *A New Logic for Electronic Commerce Protocols*. In Proc. of the International Conference on Algebraic Methodology and Software Technology, pages 499–513. Springer, 2000.

[Asokan 97] N. Asokan, Matthias Schunter & Michael Waidner. *Optimistic protocols for fair exchange*. In Proc. of the ACM Conference on Computer and Communications Security, pages 7–17. ACM, 1997.

[Bros] Warner Bros. *http://www.msnbc.msn.com/id/12694081*. In Warner Bros. to sell films via BitTorrent. Accessed on June 3, 2008.

[Catalano 04] Dario Catalano & Giancarlo Ruffo. *A Fair Micro-Payment Scheme for Profit Sharing in P2P Networks*. In Proc. of International Workshop on Hot Topics in Peer-to-Peer Systems, pages 32–39. IEEE Computer Society, 2004.

[Chaum 91] David Chaum & E van Heyst. *Group Signatures*. In Proc. of the Advances in Cryptology - EUROCRYPT, pages 257–265. Springer, 1991.

[Chaum 92] David Chaum & Torben P. Pedersen. *Transferred Cash Grows in Size*. In Proc. of EUROCRYPT, pages 390–407. Springer, 1992.

[Dario Catalano 05a] Giancarlo Ruffo Dario Catalano & Rossano Schifanella. *A P2P Market Place Based on Aggregate Signatures*. In Proc. of the International Workshop on Applications and Economics of Peer-to-Peer Systems, pages 54–63. Springer, 2005.

[Dario Catalano 05b] Giancarlo Ruffo Dario Catalano & Rossano Schifanella. *A P2P Market Place Based on Aggregate Signatures*. In Proc. of the International Workshop on Applications and Economics of Peer-to-Peer Systems, pages 54–63. Springer, 2005.

[Garcia 05] Flavio D. Garcia & Jaap-Henk Hoepman. *Off-Line Karma: A Decentralized Currency for static Peer-to-peer and Grid Networks*. In Proc. of International Conference on Applied Cryptography and Network Security, pages 364–377. Springer, 2005.

[Gehlot 07] Jai Gehlot. *Security Analysis of Two Fair Exchange Protocols*. In A thesis submitted in partial fulllment of the requirements for the degree of Master of Technology in Information and Communication Technology. DA-IICT, 2007.

[Jia 05] Zou Jia, Si Tiange, Huang Liansheng & Dai Yiqi. *A New Micro-payment Protocol Based on P2P Networks*. In Proc. of IEEE International Conference on e-Business Engineering, pages 449–455. IEEE Computer Society, 2005.

[Kai Wei 06] Alan J. Smith Kai Wei Yih-farn Chen & Binh Vo. *WhoPay: A Scalable and Anonymous Payment System for Peer-to-Peer Environments*. In Proc. of IEEE International Conference on Distributed Computing Systems. IEEE Computer Society, 2006.

[Kou 03] Weidong Kou, editeur. Payment technologies for e-commerce, chapitre Micropayments, pages 245–280. Springer, 2003.

[Micali 03] Silvio Micali. *Simple and fast optimistic protocols for fair electronic exchange*. In Proc. of the Annual Symposium on Principles of Distributed Computing, pages 12–19. ACM, 2003.

[Micropayments] Micropayments. *http://en.wikipedia.org/wiki/Micropayments*. Accessed on May 31, 2008.

[Music] EMI Music. *http://www.emigroup.com/Press/2006/press25.htm*. In EMI Music becomes the first major music company to make its catalogue available to Qtrax: the world's first ad-supported, legitimate P2P service. Accessed on June 3, 2008.

[Odlyzko 03] Andrew Odlyzko. *The Case Against Micropayments*. In Proc. of International Conference on Financial Cryptography, pages 77–83. Springer, 2003.

[Ruffo 05] Giancarlo Ruffo & Rossano Schifanella. *Scalability Evaluation of a Peer-to-Peer Market Place based on Micro Payments*. In Proc. of the International Workshop on Hot Topics in Peer-to-Peer Systems, pages 10–17. IEEE Computer Society, 2005.

[van Someren 03] Nicko van Someren, Andrew Odlyzko, Ron Rivest, Tim Jones & Duncan Goldie-Scot. *Does Anyone Really Need MicroPayments?* In Proc. of International Conference on Financial Cryptography, pages 69–76. Springer, 2003.

[Vishnumurthy 03] V. Vishnumurthy, S. Chandrakumar & E. Sirer. *KARMA: A Secure Economic Framework for Peer-to-Peer Resource Sharing*, 2003.

[Wehrheim 02] Heike Wehrheim. *Checking behavioural subtypes via refinement*. In Proc. of the IFIP TC6/WG6.1 International Conference on Formal Methods for Open Object-Based Distributed Systems, pages 79–93. Kluwer, B.V., 2002.

[Yang 03] Beverly Yang & Hector Garcia-Molina. *PPay: Micropayments for Peer-to-Peer Systems*. In Proc. of Conference on Computer and Communications Security, pages 300–310. ACM, 2003.

[Zuo 05] Min Zuo & Jianhua Li. *Constructing Fair-Exchange P2P File Market*. In Proc. of International Conference on Grid and Cooperative Computing, pages 941–946. Springer, 2005.