Practical Service Charge for P2P Content Distribution

Jose Antonio Onieva[1], Jianying Zhou[1], and Javier Lopez[2]

[1] Institute for Infocomm Research
21 Heng Mui Keng Terrace, Singapore 119613
{onieva,jyzhou}@i2r.a-star.edu.sg

[2] Computer Science Department, E.T.S. Ingenieria Informatica
University of Malaga, 29071 - Malaga, Spain
jlm@lcc.uma.es

Abstract. With emerging decentralized technologies, peer-to-peer (P2P) content distribution arises as a new model for storage and transmission of data. In this scenario, one peer can be playing different roles, either as a distributor or as a receiver of digital contents. In order to incentivize the legal distribution of these contents and prevent the network from free riders, we propose a charging model where distributors become merchants and receivers become customers. To help in the advertisement of digital contents and collection of payment details, an intermediary agent is introduced. An underlying P2P payment protocol presented in [1] is applied to this scenario without total trust on the intermediary agent.

1 Introduction

A crucial factor in the rapid growth of the Internet is electronic commerce: the ability to advertise goods and services, search for suppliers, compare prices and make payments, all being conducted at the click of a few computer mouse buttons.

Nowadays several factors have lit a fire under the peer-to-peer (P2P) movement: inexpensive computing power, bandwidth, and storage. In a P2P architecture, computers that have traditionally been used solely as clients communicate directly among themselves and can act as both clients and servers, assuming whatever role is needed at each moment. The new P2P networking paradigms offer new possibilities for content distribution over the Internet. Customer peers interchange roles with provider peers, and compete in this new networked economy. A major differentiating factor of P2P from traditional content distribution models is the lack of central management and control. This very important characteristic of P2P systems offers the ability to create efficient, scalable, anonymous - when required, and persistent services by taking advantage of the fully distributed nature of the systems.

If a peer distributing contents gets paid for this distribution, why is this peer going to distribute contents freely? This approach can incentivize a legitimate

P2P content distribution, hence avoiding the actual problems of free riders and legal issues for which P2P networks such as Napster and Gnutella have been strongly criticized [2, 3].

Popular software for P2P networking like Napster, Gnutella [4], and Freenet [5] provides everybody with opportunities to exchange low value digital goods. But potential merchants with low value goods (i.e., users inside a P2P network) have no future in such a competitive digital world due to the hardness of collection of payments and advertisement of their goods, compared with profits expected.

For such reasons, new solutions that help the merchant to gain entrance to P2P e-commerce should be designed. Previous work on paid P2P service [6] relies on a fully trusted on-line escrow server, which could be too expensive for those low value transactions. In this paper, we introduce a P2P service and payment protocol in which the load of the merchant peer is significantly reduced for only distribution of digital contents while a weakly trusted intermediary agent is used for the advertisement and collection of (small) payments.

The rest of this paper is organized as follows. In section 2, we sketch the scenario under which we envisage the distribution of digital contents inside a P2P network, and identify the security requirements in that scenario. In section 3, we describe the underlying payment mechanism used in our approach. In section 4, we present our protocol for P2P content distribution and give an informal security analysis. Finally, before concluding the paper, a more practical view of the operations needed by the peers in the content distribution is explained in section 5.

Some basic notation used throughout the paper is as follows.

- M, C, B: merchant peer, customer peer, and broker/bank, respectively
- A, TTP: agent and trusted third party, respectively
- X, Y: concatenation of X with Y
- $h(inf)$: one-way hash function over message inf
- $KeyedHash_K(inf)$: message inf is hashed using a secret key K
- $E_K(inf)$ and $D_K(inf)$: symmetric encryption and decryption of inf
- $S_U(inf)$: digital signature of entity U over message inf
- $P_U(inf)$: encryption of inf using public key of entity U
- $A \rightarrow B : X$: entity A sends message X to entity B
- $A \leftarrow B : X$: entity A retrieves message X from entity B

2 Scenario and Requirements

A market study about P2P commerce is provided in [7], where peers can find evaluation functions and results about the behavior of such a system, permitting them to make decisions in advance. In that study, different parameters are used to evaluate the market such as cost of transportation, popularity of the contents and competitiveness of the peers. In this paper, we intend to reduce costs of transportation, i.e., reduce involvement of the peers in the framework.

Figure 1 shows a general scenario we can find in a P2P application. In this scenario, each peer entity desires to earn some money by selling its files (photos, music, videos etc.). But it is hard for every entity to advertise its goods and manage the (probably small) payments with so many entities. Then such an entity can seek a purchase agent for advertising goods and collecting payments, thus it only needs to provide the digital goods/contents.

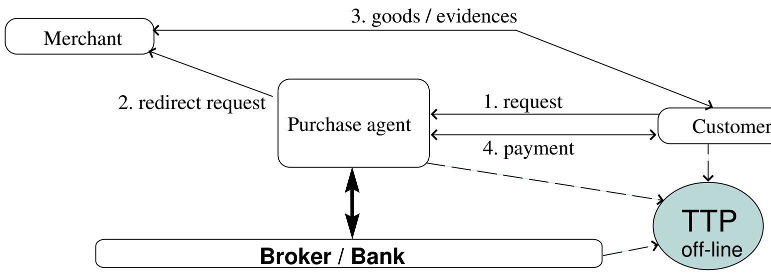

Fig. 1. P2P Service and Payment Scenario

We affirm that in a protocol where multiple entities participate, and none of them is totally trusted, i.e. collusion between any pair of them is possible, total fairness cannot be obtained. Nevertheless, we assume that the purchase agent is weakly trusted by the merchant peer in the only sense that collusion with the customer is not possible. Tools and reasons for making this type of collusion harder (although not impossible) can be found in reputation issues and incentive schemes. As an incentive for the participation of the agent in this scenario, it could earn a part of each payment or a monthly percentage of each user's successful transactions. On the other hand, collusion with the merchant peer or misbehavior of the agent by itself has to be properly and efficiently treated in our protocol.

Provision of evidence to the peers for later dispute resolution would be important to boost P2P e-commerce where exchanges are carried out between parties that probably have no prior relations and whose identities could be highly volatile.

The following properties are desirable in the above P2P service and payment scenario:

1. *Confidentiality*: The digital goods/contents should be disclosed only to the intended party (i.e. only to customers).
2. *Payer anonymity*: Payers may prefer keeping their everyday payment activities private, i.e. not allowing payees and in some cases even banks to observe and track their payments. There are two levels of anonymity: *untraceability* simply means that an adversary cannot determine a payer's identity in a run

of a payment protocol; *unlinkability* means that, in addition, participation of the same player in two different payments cannot be linked.

3. *Fairness*: The customer cannot obtain the digital goods/contents either from the intermediary agent or from the merchant unless a payment is ensured to the merchant.
4. *Timeliness*: The transacting parties always have the ability to reach, in a finite amount of time, a point in which they can stop the protocol without loss of fairness.
5. *Non-repudiation*: It is impossible for a sender peer, after a successful execution of the protocol, to deny having distributed the digital goods. It is impossible for an agent, after a successful execution of the protocol, to deny having received the payment.
6. *Light-weight merchant*: Since the protocol is run in a P2P scenario, merchant peers should not be overloaded with payment issues.

3 A P2P Payment Protocol

General purpose electronic payment systems have been widely studied, which can be classified into two categories: *cash-like* systems and *check-like* systems. In cash-like systems, special tokens denominated as electronic coins (or cash) are used [8, 9]. The payer has been previously taken away an amount of money in a withdrawal protocol, hence they are *pre-paid* payment protocols. In check-like systems, the payer usually issues a form (whether it be a check or a credit card slip) to the payee [10]. There is no previous withdrawal of money, and the payee must ensure that the payer possesses enough money to carry out the payment. So consulting the payer's bank is necessary prior to accept it. Such systems are also denominated as *on-line* verification payments (e.g. [11]).

An electronic payment system for P2P scenarios was proposed in [1]. In this protocol, three entities are involved: merchant, customer, and broker/bank who is trusted by the other entities. The same notation used in the original paper is listed below for the understanding of this scheme.

- ID_X: identity of entity X
- K_X: secret key used by entity X (and known only to it)
- $SerNum$: unique serial number associated with every digital note
- $Value$: value associated with the digital note
- $T0, T1, T2, T3$: time of issue, deadline of redemption, start time of refund, and expiry time for the digital note, respectively

Each digital note is prepared by the merchant as follows.

- $ID_{Material} = ID_M, Value, SerNum, T0, T1, T2, T3$
- $DigitalNote = ID_{Material}, KeyedHash_{K_M}(ID_{Material})$

The main advantage of having the merchant creating its own digital notes is that it can check double spending *before-the-fact* without contacting the broker/bank. The merchant peer then transfers digital notes to the broker who adds a broker stamp such that a stamped digital note format is [*DigitalNote, BrokerStamp, SVC*].

- $BrokerStamp = ID_B, KeyedHash_{K_B}(DigitalNote, ID_B)$
- Stamp Verification Code (SVC) = $h(BrokerStamp)$

Once the broker transfers SVC to the merchant, the stamped digital note is ready for circulation. Whenever the customer peer wants to purchase goods from a particular merchant peer, he approaches the broker, obtains a certain amount of digital cash (issued by that merchant peer and stamped and stored by the broker) using macro payment mechanisms such as credit card payment schemes. The stamped unspent digital note should be kept secret and protected by the entity possessing it, either by the broker or by the customer.

A merchant peer can redeem the value of digital cash before time T1 associated. The merchant peer has to reveal the broker stamp to the broker. If the broker stamp is valid, the broker credits the merchant peer's account and marks the digital note as spent. Similarly, the customer peer can refund its unspent digital note before expiration, that is, after time T2 but before time T3.

For the transaction between the merchant and customer peers, the customer sends the stamped digital note excluding the broker stamp to the merchant, who verifies that this SVC is unspent, and that current time is less than T1. Then, a fair exchange protocol is assumed for the exchange of the digital good and the broker stamp. Although the original paper did not provide any detail about the fair exchange, we claim that all the parties involved should have the ability to check the broker stamp validity.

4 Our Approach

We design a P2P content distribution and payment protocol in which the load on the peer that plays the role of merchant is significantly reduced, hence motivating the participation of peers in this type of evolving e-commerce. The basic idea is the delegation of the merchant role to the agent during the payment phase. With this change, the customer peer does not need special digital notes for each merchant peer. Instead, he can buy digital contents from several merchant peers by interacting with only one agent who represents these merchant peers. Thus, the payment view of the P2P network changes to a new model (see Figure 2).

A prior notation needed for the complete understanding of the protocol is as follows.

- *DigitalContent*: digital content that the merchant peer M sells to the customer peer C
- *descr*: description of the digital content that C obtains before starting purchase (i.e. from the agent's web site)
- P_{ID}: identifier of the digital content and its price
- *utsn*: unique transaction serial number
- $L = (utsn, P_{ID})$: label of the current transaction

- k_c: session key generated by the agent and used by M to encrypt the digital content
- $Cipher = E_{k_c}(DigitalContent)$: ciphertext of the digital content encrypted with k_c
- $dc = h(DigitalContent)$: digest of the digital content
- $IntegritySign = S_M(dc, descr, P_{ID})$: digital content verification code generated by M and available at the agent's web site
- $t = P_{TTP}(A, M, k_c)$: ciphertext of the session key encrypted with the *TTP*'s public key

4.1 P2P Service and Payment Protocol

In our protocol, we assume that each peer (acting as a customer) can set up a secure and confidential channel (SSL or IPSec) with its agent, broker/bank, and the *TTP*, and the agent can also establish such a channel with the broker and the *TTP*. Our protocol consists of a main protocol and two sub-protocols. In the normal situation, only the main protocol will be executed among the customer peer, the agent, and the merchant peer, while the *TTP* is off-line and not involved. If there is something wrong in a transaction, the agent can initiate the *cancel* sub-protocol and the customer peer can initiate the *resolve* sub-protocol to terminate the transaction without loss of fairness.

The agent will prepare the digital notes for the merchants that it represents, and send these digital notes to the broker for stamping. The customer can obtain the stamped digital notes from the broker using a macro payment mechanism (e.g. credit card), and use these digital notes in purchase of digital goods/contents from a merchant (via its agent).

Suppose the customer C has obtained some digital notes from the broker B. At the beginning, C accesses information in the agent A's web page, and downloads $descr$, P_{ID}, and $IntegritySign$. Then C launches the following P2P service and payment main protocol.

$$\begin{array}{ll}
1.\ C \rightarrow A & : M, L, DigitalNote, SVC \\
 & \quad A \text{ checks and IF correct follows} \\
2.\ A \rightarrow M & : M, L, k_c, t, SVC, S_A(M, L, k_c, t, SVC) \\
3.\ C \leftarrow M & : A, L, Cipher, dc, t, S_M(A, L, h(Cipher), dc, t, SVC) \\
4.\ C \rightarrow A & : BrokerStamp \\
5.\ A \rightarrow C & : M, L, k_c, S_A(M, L, k_c)
\end{array}$$

At Step 1, the customer C sends a digital note to the agent A. A makes all necessary checks on the digital note before notifying the merchant M at Step 2 that there is a request pending from C. Such checking includes that the current time is earlier than the deadline of redemption T1, and that the digital note has not been spent yet. If correct, A provides M with its signature which could be used to prove the amount of payment to be credited to M's account (if the transaction is completed) and the session key for encryption of the digital content. A also encrypts the session key k_c with the *TTP*'s public key (in order to reduce the computational load on the merchant peer host). After verifying the purchase request redirected by A, M prepares the encrypted digital content and its signature which could be used to prove the origin of the digital content. C retrieves the encrypted digital content from M at Step 3. Prior to submit the broker stamp to A at Step 4, C verifies whether M is committed that the digital content sent in ciphertext is the one as C expected. A releases the session key at Step 5 after obtaining the valid broker stamp from C.

If the above protocol is executed successfully, the customer peer obtains k_c for the decryption of $Cipher$, and thus the digital contents, and the agent obtains the broker stamp. Then the agent can send the broker stamp to the broker for redemption.

If A does not receive the broker stamp in a pre-determined amount of time before T1, it can launch the following *cancel* sub-protocol.

$$\begin{aligned}
4'.\ A \rightarrow TTP &: A, M, L, SVC, S_A(cancel, A, M, L, SVC)\\
&\quad \text{IF not resolved THEN}\\
5'.\ A \leftarrow TTP &: S_{TTP}(cancel, A, M, L, SVC)\\
&\quad \text{ELSE}\\
5'.\ A \leftarrow TTP &: BrokerStamp
\end{aligned}$$

In such a case, A sends to the TTP a cancel request. Then, if the protocol has not been resolved the TTP verifies A's signature on the request. If correct the TTP signs a cancel affidavit. If the protocol was resolved by C, the TTP gives A access to retrieve the valid broker stamp. Note that the agent can obtain the broker stamp and a cancel affidavit, which result in an unfair situation. Nevertheless we consider a *poll* solution to revoke the broker stamp redemption. In this poll solution, the broker has access to the cancel affidavits from the TTP server and then it will execute that operation (searching for fraudulent redemption operations) before redeeming the agent.

If C does not get the session key for decryption of the digital content in the main protocol before time T1, it appeals to the TTP in a *resolve* sub-protocol.

$$\begin{aligned}
5'.\ C \rightarrow TTP &: A, M, L, h(Cipher), dc, t, SVC, S_M(A, L, h(Cipher), dc, t, SVC),\\
&\quad BrokerStamp\\
&\quad \text{IF not cancelled THEN}\\
6'.\ C \leftarrow TTP &: k_c\\
&\quad \text{ELSE}\\
6'.\ C \leftarrow TTP &: S_{TTP}(cancel, A, M, L, SVC)
\end{aligned}$$

In such a case, C sends to the TTP all the information received from M as well as the broker stamp. If the protocol has not been cancelled, the TTP verifies M's signature and checks whether the hash of the broker stamp equals SVC. If everything is positive, the TTP decrypts t, verifies that the key k_c is intended for A and M, and finally stores k_c for C's access. If the protocol has been revoked by A, the TTP will provide a cancel affidavit.

Some financial issues should be taken into account. The agent could send all broker stamps to the broker in batch mode. Similarly, the broker could credit the agent account in batch mode, giving an elapse time for this operation, such that it can retrieve from the TTP all the cancel affidavits and revoke the broker stamp redemption (and hence the agent's bank account credit operation) if needed. None of these financial assumptions seems to be hard to obtain.

Finally we would like to state that a complementary design based on a reputation system [12] could help to boost the P2P commerce. Reputation is the only mechanism available to peers in order to evaluate a candidate provider of a requesting service in terms of quality, reliability and correctness and thus plays significant roles in the selection of agents. So, if a situation arise in which an agent is misbehaving, a network of reputation can "mark" this entity thus preventing the next fraudulent action.

4.2 Dispute Resolution

Disputes can arise, and we show how the resolution with an arbitrator proceeds for all the entities involved in such a dispute.

Origin of digital content: If M denies having sent a particular digital content, then C gives $descr, IntegritySign, A, L, P_{ID}, Cipher, dc, t, k_c, SVC$, and M's signature to the arbitrator. The arbitrator checks

- $descr$ fits with $DigitalContent$
- $dc = h(DigitalContent)$
- $IntegritySign$ is M's signature on $(dc, descr, P_{ID})$
- $t = P_{TTP}(A, M, k_c)$
- $DigitalContent = D_{k_c}(Cipher)$
- M's signature on $A, L, h(Cipher), dc, t, SVC$

If all the above checks are positive, the arbitrator concludes that the digital content is from M. If C receives a wrong digital content, some of the first three checks in the list might be false. However, C can demonstrate the misbehavior of M with $IntegritySign$. If M can present A's signature on a different session key k_c for the same transaction L, the arbitrator concludes that A is the misbehaving party.

Payment received by A: A possible dispute could arise between M and A if the latter did not credit M's account after transferring a broker stamp to B for redemption. A could obtain incentives from the merchant peers, and the commission depends on how many successful payments it carries out. However, A may try to keep the entire payment of the digital good. If A denies having completed a transaction (L, SVC), M should present to the arbitrator M, L, k_c, t, SVC and A's signature on it. Then the arbitrator checks the signature and if A cannot present a cancel affidavit signed by the TTP for that transaction (L, SVC), the arbitrator concludes that A completed the transaction and must pay M for it. Note that if A tries to misbehave by completing the transaction and obtaining a cancel affidavit it will eventually succeed. But B will prevent it from crediting A's account if B obtains the cancel affidavit from the TTP. In this case only

C will be benefited. C obtains the k_c and thus the digital content while A is not redeemed. C could spend the same stamped digital note later again, or get refunded from B.

Invalid broker stamp: If C tries to misbehave by sending an invalid broker stamp, two cases are possible.

- C sends the invalid broker stamp at Step 4. Assume that A is not going to collude with C as we discussed in section 2, A will detect an invalid broker stamp using SVC and will reject it.
- C stops the protocol at Step 4, and contacts the TTP to resolve. If the transaction has not been cancelled, the TTP will check with SVC signed by M that the broker stamp provided by C is valid before providing k_c to C.

Origin of SVC: If M colludes with C and sends to C at Step 3 a SVC which has already been spent by C. Then C could contact with the TTP and successfully get k_c with the resolve sub-protocol. Whenever A tries to fetch the broker stamp from the TTP, it will discover M's fraudulent behavior and go to the arbitrator. If M can present an A's signature on the same transaction (L, SVC), the arbitrator concludes that the misbehaving party is A and A will have to pay for the transaction to M. Otherwise, M is identified as the colluding party.

4.3 Security Analysis

Now we informally analyze whether our P2P service and payment satisfies the requirements described in section 2.

- *Confidentiality*: The digital goods/contents are disclosed only to the intended customer peer. Although the agent knows the deciphering key, it does not have the knowledge about the encrypted digital content if it is transmitted over a private channel (e.g. SSL or IPSec enabled) from M to C. Similarly, the TTP, if involved, cannot get the encrypted digital contents either.
- *Payer anonymity*: Only the first level of anonymity is reached, that is, untraceability. A customer peer never needs to reveal its identity except its IP address for receiving messages during the protocol.
- *Fairness*: As we mentioned before, our fairness is achieved under the assumption of no collusion between the agent and the customer. In the main protocol, fairness will not be lost until Step 3 since neither the agent has obtained the broker stamp nor the customer gets the key to decrypt the digital content. After Step 3, both of them obtain what they expect (digital good and broker stamp) or none of them obtains any valuable information.
- *Timeliness*: After notifying the merchant of a request for purchase at Step 2, the agent has the ability of cancelling the protocol if needed to reach the end of the protocol without breach of fairness. On the other hand, the customer can terminate the protocol at any time before releasing the broker stamp, or initiate the resolve sub-protocol after Step 4.

- *Non-repudiation*: Proofs of origin of digital contents and payment received by the merchant are discussed in section 4.2. If the digital content provided at the end of a successful execution of the protocol does not fit with the description signed by the merchant on $IntegritySign$, the customer can obtain the evidence from Step 3 for a dispute resolution. If the agent cheats the merchant by falsely denying receipt of the payment from the customer, the merchant can get the evidence from Step 2 for a dispute resolution.
- *Light-weight merchant*: For each protocol run, the merchant only needs one signature verification and generation. (Although $IntegritySign$ token is also generated by the merchant, this operation can definitely be carried out in an off-line process.) More importantly, the merchant only receives one service request from the agent, and makes the encrypted digital contents available to the customer. The merchant does not need to take care of advertisement and payment.

5 Practical View

In order to give a more practical view of the involvement of the entities in our P2P service and payment protocol, we give an instantiated execution of the protocol. We define a typical P2P scenario where one of the peers tries to purchase a file. This file description and associated advertisement are hosted in an agent server. Note that if an agent advertises similar files (belonging to different peers), an analysis about the competition between different peers in the distribution of the contents should be undertaken. A preliminary study can be found in [7].

We sketch a scenario (see Figure 2) where a previous contract or relation exists between a merchant peer and an agent. This is something totally necessary, since at least, the merchant must register to use the agent's hosting services. As we analyzed earlier, the merchant peer has a very light participation in the protocol, an important property that will facilitate the involvement of peers distributing, in an exchange for a small amount of money, digital contents over P2P networks.

1. A peer, who is surfing the web, visits *http://www.curious-papers.com* and once inside, clicks the section "Snakes". He reads the abstract or description *descr* and decides to buy it. So he pushes the button "buy it". This operation forms a transaction label L, downloads the content verification code $IntegritySign$ token, and uploads a valid stamped digital note (excluding broker stamp).
2. The agent's server verifies the validity of the digital note, and checks that SVC is unspent. If correct, it redirects this request to another peer who owns the paper, along with the product information received from the customer peer (L), the key needed to encrypt the contents (k_c), and the fingerprint of the broker stamp (SVC).
3. The merchant peer prepares the encrypted version of the paper (*Cipher*), and generates a signature. Then it notifies the customer peer to retrieve.

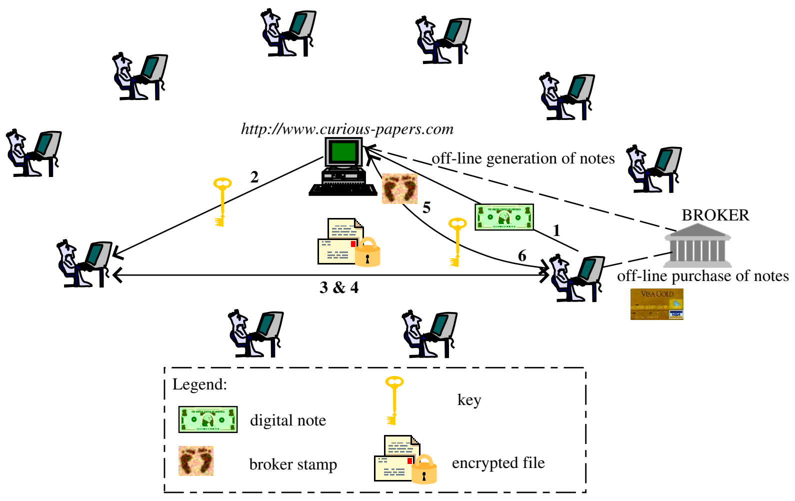
http://www.curious-papers.com
off-line generation of notes
2
5
1
6
BROKER
3 & 4
off-line purchase of notes
Legend:
key
digital note
broker stamp
encrypted file

Fig. 2. Application Scenario

4. The customer peer downloads the cipher paper and the merchant peer's signature. An add-in component in the customer peer's browser verifies the merchant peer's signature and the fingerprint of the broker stamp. If correct, the customer peer is asked for approval of the description signed by the merchant peer by pressing the button "OK".
5. Then the customer peer's computer sends the broker stamp to the agent. A window of notification should pop up to advise the customer that once the broker stamp is sent, it will be the non-return point of the transaction. The add-in component waits for the session key.
6. After receiving the broker stamp, the agent proceeds to send the session key. At the customer peer side, the add-in component will verify the agent's signature, decrypt the cipher paper, and display the paper to the customer.
7. If the agent does not receive the broker stamp within a determined time (depending on the security policy) a cancel sub-protocol can be launched, obtaining either a cancel affidavit for the digital note or the valid broker stamp from the TTP.
8. If the customer peer does not receive the session key within a determined time, the add-in component redirects a request to the TTP in order to resolve the protocol. If the session key is received, the add-in component will decrypt the cipher paper and display the paper to the customer. If a TTP signed cancel affidavit is received, the add-in component will pop up a window to notify the customer that the transaction has been cancelled.

Redemption and refund phases proceed according to the underlying P2P payment protocol. The broker has the ability of cancelling the redemption phase as stated before.

6 Conclusion

With the emergence of wireless technology, grid computing, and other technologies where the storage and transmission of data are carried out without a centralized server, it is clear that new models of charging and distribution should not only comply with the requirements of this new topology but also provide with an efficient and practical solution. In this paper, we introduced a new entity that without being totally trusted, acts as a hub of the topology, helping the distributors in collection of possibly small payments and advertisement of the digital contents. We made use of an underlying P2P payment protocol and applied it to our practical P2P content distribution scenario where a merchant peer's workload is largely shifted to an intermediary agent thus each peer can be easily involved in distributing digital contents and receiving payment via the agent. We also discussed the trustworthiness presumed to each of the entities in our model.

References

1. Anantharaman, L., Bao, F.: An efficient and practical peer-to-peer e-payment system. manuscript (2002)
2. Adar, E., Huberman, B.: Free riding on gnutella (2000)
3. Golle, P., Leyton-Brown, K., Mironov, I., Lillibridge, M.: Incentives for sharing in peer-to-peer networks. Lecture Notes in Computer Science **2232** (2001) 75–87
4. http://www.gnutella.com.
5. http://freenet.sourceforge.net.
6. Horne, B., Pinkas, B., Sander, T.: Escrow services and incentives in peer-to-peer networks. In: Proceedings of the 3rd ACM conference on Electronic Commerce, ACM Press (2001) 85–94
7. Antoniadis, P., Courcoubetis, C.: Market models for P2P content distribution. In: AP2PC'02. (2002)
8. Boly, J.P., Bosselaers, A., Cramer, R., Michelsen, R., Mjolsnes, S.F., Muller, F., Pedersen, T.P., Pfitzmann, B., de Rooij, P., Schoenmakers, B., Schunter, M., Vallee, L., Waidner, M.: The ESPRIT project CAFE - high security digital payment systems. In: ESORICS. (1994) 217–230
9. Rivest, R.L., Shamir, A.: Payword and micromint: Two simple micropayment schemes. In: Security Protocols Workshop. (1996) 69–87
10. Asokan, N., Janson, P.A., Steiner, M., Waidner, M.: The state of the art in electronic payment systems. IEEE Computer **30** (1997) 28–35
11. Bao, F., Deng, R., Zhou, J.: Electronic payment systems with fair on-line verification. In: IFIP TC11 16th Annual Working Conference on Information Security: Information Security for Global Information Infrastructures, IFIP TC11, Kluwer Academic Publishers (2000) 451 – 460
12. Damiani, E., Vimercati, S.C.D., Paraboschi, S., Samarati, P., Violante, F.: A reputation-based approach for choosing reliable resources in peer-to-peer networks. In Atluri, V., ed.: Computer and Commmunications Security, ACM (2002) 207–216