BARTER: a Backbone ARchitecture for Trade of ElectRonic content*

Gadi Shamir, Michael Ben-Or, and Danny Dolev

Institute of Computer Science
The Hebrew University of Jerusalem, Jerusalem, Israel

Abstract. BARTER is a scalable, highly-available and efficient electronic commerce system that facilitates digital content trade over an open network. BARTER is designed to operate over a large-scale, global and heterogeneous communication network infrastructure. The BARTER protocols address two vital requirements from an electronic commerce system, neglected from existing systems: *scalability* and *transactional efficiency*. BARTER suggests three basic protocols, offering different levels of trust in the system.

BARTER's novelty is twofold: First, by distributing signature verification away from BARTER servers, the overhead of online transaction processing is reduced by orders of magnitude. Nevertheless, this does not sacrifice strong security requirements, namely the ability to aggregate transaction commitments and to resolve disputes.

Second, BARTER integrates scalability considerations into several system components that are likely to suffer service degradation in a world-wide setting. These components are the authentication subsystem, the account management subsystem, and the maintenance of global data (such as transaction identifier lists). In addressing these issues, BARTER takes into account the inherent asynchronous, unreliable, insecure and failure-prone environment assumptions. We contend that by employing service distribution, BARTER is expected to scale well, meeting the demands of a world-wide setting, over which it is intended to operate.

1 Introduction

The amazing growth of the Internet has brought the realization that cyberspace is an outstanding platform to conduct business: high connectivity, global presence, ever-growing population, and an element of trend constitute a perfect mixture, creating a demand for mechanisms that enable to buy and sell goods and services online.

The development of scanning, OCR and information retrieval techniques, hand in hand with a highly-connective open network yielding large bandwidth, leads to a common belief in the flourishing future of an information goods market. This market is expected to overwhelm many traditional forms of commerce. When referring to the electronic market, most suggestions visualize an economy

* A patent application on this work is being filed

of *digital* goods and services. Any party that can benefit financially from offering electronic content is considered a potential vendor.

An *electronic transaction* can be defined as an exchange of digital content between two or more parties. Digital content, in this sense, is any valuable string of bits: articles, web pages, digital library queries, images, video programs, tokens, and contracts (including 'informal' commitments to deliver goods or services).

Though trading in such an atmosphere suggests economical and intellectual prosperity, it requires commerce mechanisms that would enable intellectual property owners to offer their digital content for sale. Such mechanisms must address some inherent characteristics of open-network commerce: a massive population of shoppers of vendors, in which long-term customer-merchant relationships are scarce, and a low level of trust between customers and merchants. These require that an electronic commerce (EC) system should satisfy a diverse list of requirements: it should be widely acceptable, allow alternate payment methods, be easily deployable and integrable, and should comply with regional laws. Beyond these requirements, most suggestions focus on issues such as security, reliability, scalability and high availability, transactional efficiency, privacy, and auditability.

The last few years have brought many proposals for electronic commerce systems and EC protocols that address a wide variety of requirements and adjust to a varying set of assumptions. These systems fall into three general categories - secure credit card systems [6, 22], token systems [15, 23, 2, 8], and notational (credit-debit) systems [1, 16, 26] (this categorization is borrowed from [11, 12]).

Our work focuses on the third category, notational (credit/debit) systems. Notational money exists as entries in the ledgers of some financial institution. Notational systems use aggregation to eliminate transactional-level dependency upon existing financial systems. The ledger holder and the transaction processing gateway are not necessarily the same physical entity. Merging these functionalities, however, may increase transaction processing efficiency. This lends itself to the idea of a trusted broker that acts as a mediator between customers and merchants, performing *fair* exchange of the traded items.

In this work, we present BARTER: a Backbone ARchitecture for Trade of ElectRonic Content. BARTER's basic functionality is providing means of exchanging digital content over an open network. To the best of our knowledge, BARTER is the first EC system to address transactional efficiency and scalability issues, without sacrificing other important requirements such as transactional reliability, fair exchange of traded items, and support for evidence collection and arbitration. We suggest three basic protocols, offering various levels of trust in BARTER. We do not assume the occurrence of special purpose hardware, such as observers [14]. This kind of equipment requires the creation of a large infrastructure (reader interface in every computer), and is thus assumed to play an important role in retail commerce, and not in open-network commerce. The next three paragraphs highlight BARTER's benefits over existing EC systems.

Cross-verification of signatures. Strong security properties of an EC system are usually achieved using cryptographic primitives such as digital signa-

tures. Parties commit to their actions by producing undeniable proofs attesting to their consent. Digital signature verification is an expensive operation in terms of computational resources[1]. In BARTER, the parties involved in a transaction *cross-verify* their commitments. Recently, "optimistic" exchange protocols have been suggested [4, 5], according to which a trusted party intervenes in the protocol only when one of the parties misbehaves, or in case a failure hinders protocol completion. In BARTER, payment clearing commitments are typically submitted online, however, signature verification by a BARTER server is redundant. In fact, by performing simple string comparison operations during transaction processing, few symmetric key operations, and several database queries/updates, BARTER may be utilized for agreement validation, while commitment authenticity is assured. Consequently, the load on BARTER servers is significantly low, and can, in practice, yield a very high transaction rate.

Distribution of account maintenance. Some existing EC systems assume a central server or broker through which all transactions are processed and by which all accounts are maintained. In light of the expected swelling user base, such solution is likely to become a bottleneck and suffer a storage boom. In BARTER, service distribution is implemented by a group of servers which balance the system load and storage requirements.

Distribution of the authentication subsystem and global data. Many electronic commerce systems base their protocols on various global clock synchronization assumptions. These assumptions are sometimes required for correct implementation of an authentication subsystem (see Section 2), or for detecting payment commitment replay (see Section 4). In practice, keeping a network of servers synchronized at a global scale is a difficult problem. BARTER protocols do not assume global clock synchronization. Instead, efficient bookkeeping methods are used to detect replay attempts and to keep track of the authentication subsystem tokens.

This paper is organized as follows: Section 2 gives a brief overview of BARTER, specifies the environment and model assumptions, and sketches the authentication subsystem. Section 3 surveys related work. Section 4 outlines the BARTER protocols. A complete description and a thorough analysis can be found in [31]. Section 5 presents a brief analysis of BARTER. Section 6 concludes and gives suggestions for future research.

2 Overview of BARTER

BARTER is designed to operate over a large-scale, global, and heterogeneous communication network infrastructure, such as the Internet. Users of BARTER are providers and consumers of electronic content - *merchants* and *customers*, respectively. Note that the terms merchants and customers imply a special kind of exchange, in which electronic goods are traded for money. Nevertheless, simple modifications to the BARTER protocols allow for other types of exchange, such as

[1] Assuming a trapdoor-function implementation of digital signatures such as RSA [30] or DSA [25]

simultaneous exchange of messages or contract signing. In general, any type of exchange in which the traded items are valuable to both sides is feasible. Henceforth, we will use the term *users* to denote merchants and customers. BARTER services are provided by a group of servers, which span over a large number of geographically distant network locations. The servers and users communicate by sending messages through the underlying network.

An *account* is established and maintained for each BARTER user. Accounts are partitioned into logical *realms* (which will most probably correlate to geographical realms). Transactions processed by BARTER essentially invoke exchange of digital content between users of the system, through the intervention of one or more BARTER servers. Where digital content is exchanged for payments, BARTER maintains a monetary balance for each user; the item price is then settled among the corresponding accounts involved in a transaction[2].

Each realm is uniquely identified by a name, and is managed by a BARTER *realm server*. Realm servers are the network representatives of BARTER: they are responsible for maintaining realm accounts and processing transactions relating to their realm. Realm servers have well-known network addresses. The BARTER server infrastructure might be operated and maintained by a large financial institution. Transaction fees, either temporal (e.g., a monthly subscription) or by volume (e.g., fixed price or percentage per transaction), serve as an incentive for the institution to provide trading services.

BARTER supports three basic transaction protocols, each of which assumes a different level of trust in BARTER, and possesses different properties:

Light-Weight Exchange **(LWE)**: The LWE protocol does not require the parties to generate nor verify commitments, and is therefore expected to be highly efficient in terms of transactional overhead.

Remote Commitment Verification **(RCV)**: The RCV protocol requires that *users* of BARTER generate and cross-verify signed commitments. BARTER is *not* required to perform online verification, but rather ensures the existence of an agreement between the parties, and stores the parties' commitments.

Asymmetric exchange - RCV **(AS-RCV)**: This protocol assumes the special case in which payment is traded for digital content, and allows the parties to remove their trust in BARTER for storing and supplying commitments. It incurs an asymmetric message flow between the parties, and requires them to retain the corresponding commitments in long term storage.

All BARTER protocols guarantee several important properties:

Delivery atomicity: This property (sometimes referred to as *fair exchange* or *goods atomicity*) ensures that in a transaction involving two items t_1 and t_2,

[2] Means of interoperability with the existing financial system must be defined, allowing users to transfer funds to and from accounts held by the system.

meant to be exchanged by parties p_1 and p_2, p_1 is delivered item t_2 if, and only if, p_2 is delivered item t_1. This notion can be easily extended to a transaction involving n items and parties.

Agreement validation: The parties involved in a transaction must agree over its details.

Content arbitration: Content complaints occur when some of the parties involved in a transaction are dissatisfied with the nature and/or quality of the items delivered to them. Such complaints can be resolved by filing a complaint, along with the proper evidence, to an appropriate (possibly human) arbitrator.

The **RCV** and **AS-RCV** protocols satisfy, in addition to the above mentioned properties, two other properties: they eliminate the possibility of BARTER forging transactions, and they allow for the resolution of transaction denial complaints (see Section 5). The term denial complaints refers to a situation in which users claim they have not given their consent to a specific transaction, although they have learned from BARTER of the existence of it. Alternatively, users may argue that they *have* given their consent, but to a transaction whose details are different than the one that was processed.

2.1 Model and Assumptions

BARTER assumes network layer services are unreliable and insecure. Messages sent through the network may be lost, or worse - intercepted, analyzed, tampered with and easily reintroduced by eavesdroppers. Further, the network may partition, detaching different components, which would then be unable to communicate due to this failure.

The environment is assumed to be *asynchronous*: messages may be delayed in transit for an arbitrarily long time. The system servers and users have access to local clocks, however it is not assumed that these clocks are synchronized.

We further assume that users of BARTER may deviate arbitrarily from their prescribed protocol, in attempts to commit fraudulent actions, or actions that interfere with the system's normal functioning. Such behavior is commonly referred to in the literature as *byzantine failures* [20].

Third parties may also interfere with the system protocols, either passively, by eavesdropping on the protocol communication, or actively, by timely introducing old or fabricated messages into a session of a system protocol. Though denial of service attacks are not dealt with explicitly, note that such attacks are likely to limit service availability only at a specific domain, and therefore have only a minor effect on the overall system availability.

We assume BARTER servers are physically secure, and that they do not deviate from their prescribed protocols. Nevertheless, the possible outcomes of an invalidation of these assumption are not disastrous: users may retain non-repudiable evidence for actions they committed to, and therefore corrupted actions of a system server or servers can be unwinded.

Despite this, BARTER servers may fail *benignly* (stop-fail), even during the process of executing a single transaction.

2.2 BARTER Authentication Subsystem - A Scalable Approach

BARTER adopts the realm approach to authentication, motivated by the well-known Kerberos system [19, 18]. Authentication realms in BARTER overlap account realms. In the typical case, users may authenticate directly to their BARTER realm server (e.g., by using a certified public key at an initial authentication stage [34], and caching a short-term session key at the server). Cross-realm authentication is achieved by establishing a long-term shared symmetric key between each pair of BARTER servers. Note that in general, cross-realm authentication can be problematic, since it may force competing entities to enter into unreasonable trust relationships, and incurs complex organizational and bilateral considerations. However, in a single organization offering a confined service, such as BARTER, cross-realm certification is feasible. When transaction processing crosses different realms, BARTER offers three approaches:

Direct authentication to a remote realm, using a public key certified by BARTER. This method requires some certificate revocation mechanism. Most likely, realm CRLs (certificate revocation lists) will be maintained by BARTER realm servers.

Indirect authentication to a remote realm, using realm tickets, similar in form and nature to Kerberos tickets. Such tickets can be obtained at a BARTER realm server. In Kerberos, a lifetime field in the ticket prevents its replay after expiration. This method requires perfect clock synchronization [10], a dubious assumption in a wide-area setting. BARTER implements a different approach, which requires some bookkeeping: issued tickets lists are maintained at every BARTER server, and garbage collection is carried out in the background through a 'gossip' process.

Local authentication, according to which the users involved in a transaction authenticate to their *local* realm server; BARTER servers settle the corresponding transaction through a process which will be described in the Section 4. We assume this will be the typical case.

3 Related Work

Secure credit card transaction systems constitute a major class of suggested electronic payment protocols. The idea behind secure credit card systems is utilizing an existing world-wide infrastructure and customer base. Traditional credit card systems clear up disputes by contractual means, i.e., by offering an insurance policy that compensates for losses resulting from fraud. In addition, bad-behaving customers and merchants are tracked and rated. The *iKP* protocol family [6] and its successors (e.g., SET [22]) attempts to solve this problem by requiring merchants and customers to commit to their transaction by digitally signing each payment order and submitting the payment orders to a payment

gateway. This gateway, sometimes referred to as the acquirer gateway, serves as an interface between existing credit card clearing systems and electronic remote-payment protocols. Commitment signatures are processed by this gateway and therefore it constitutes a potential bottleneck; Moreover, each and every transaction (assuming no merchant-specific aggregation) is processed both by the payment gateway *and* the existing inefficient, bottlenecked transaction clearing infrastructure. Consequently, secure credit card systems are an outstanding opportunity for gradual deployment of network payment services; Yet, once a large enough customer base becomes part of a certified public key infrastructure, credit card numbers are rendered obsolete, and a network of payment gateways can be utilized for executing improved protocols, such as BARTER, satisfying superior properties.

In a different vein, cryptographic techniques such as digital signatures and blind digital signatures [13] allow for perfect unforgeability and perfect untraceability of digital tokens, and therefore form a theoretical foundation for intriguing research in decentralized, untraceable transactions. Yet, an inherent property of digital tokens is that they are easily copied. Thus, electronic token instruments must apply copy detection or prevention mechanisms. Some alternatives are on-line token validation, which may cause a bottleneck, or use of special-purpose hardware for witnessing and authorizing token spending, which requires the installation of special-purpose hardware. In a classic protocol [15], double spending is detected at the deposit phase. However, this comes at the expense of an impractical cut-and-choose protocol executed in the withdrawal phase.

Perhaps the most serious technical drawback of untraceable electronic currency protocols is their high vulnerability to network omission failures. This can be exploited for fraud, a problem identified and tackled at [33, 35, 11, 32]: A dishonest party, be it the merchant receiving a token as payment or the customer withdrawing a token, can deny its receipt, exploit the token's anonymous properties and then spend it or redeem its value.

In addition, the notion of untraceability is criticized for being socially and legally imperfect. Anonymous untraceable actions, especially when anonymity is unrevokeable, can be abused in crimes such as blackmail or money laundry [36]. In fact, in the United States, the Money Laundry Act (§12 USC 1829) requires that EC systems report and record high-valued transactions [35].

Moreover, anonymous electronic tokens involve expensive cryptographic protocols, and therefore might be inefficient in practice[3]. Later research suggests an improvement in efficiency (e.g., [9]).

Microtransaction systems (e.g., [21, 29, 27]) are inspired by a presumption, that many types of information goods can be decomposed, without incurring extra cost, to extremely low-valued items. The inherent cost of these items is assumed to cover not only the value of the items themselves, but also the filtering and re-bundling performed on these items to meet specific demands. The idea here is reducing the use of computationally-intensive cryptographic primitives, thereby abandoning some per-transaction security objectives. Nevertheless, it is

[3] Tygar [35] estimates the cost of each transaction to be approximately one dollar.

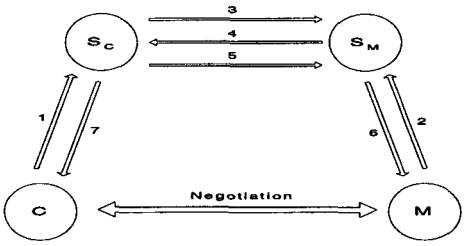

Fig. 1. BARTER Protocol Message Flow: Messages 1 and 2 indicate submissions of *CTC*s to BARTER; Messages 3, 4 and 5 indicate internal transaction processing by BARTER; Messages 6 and 7 indicate the transaction result messages

argued that the cost of breaking the protocol is higher than the value of the transaction anyway.

When the traded objects are of electronic nature, strict delivery assurance and dispute resolution techniques can be employed. A major step in this direction was taken in NetBill [16]. The NetBill system, which has inspired the development of BARTER, assumes the existence of a global, trusted server which maintains accounts for customers and merchants. Each NetBill transaction is processed by the NetBill server. In [35], the NetBill protocol is shown to provide several plausible properties, such as *goods atomicity* (money is transfered if and only if goods are delivered) and *certified delivery* (each party can present non-repudiable proofs of what was delivered and what was sent). Achieving these properties requires the NetBill server to verify two digital signatures for each transaction. Hence, the NetBill server is apt to become a processing bottleneck, and transaction capacity is in practice severely limited.

4 BARTER Protocols

The basic BARTER protocol message flow is shown in Figure 1. BARTER protocols consist of three phases: negotiation, commitment and processing. Recall that we have suggested three optional versions of the protocols. These versions differ in the commitment and processing stage, as will be explained below. In this section, the protocols are briefly overviewed. A complete description and a thorough analysis can be found in [31].

We assume the transaction involves two parties, a customer and a merchant. As already mentioned, although our discussion includes 'traditional' customer-merchant interaction terms, where a digital item is sold for an agreed-upon price, our protocols are oriented toward general digital item exchange, which could involve more than two parties.

Henceforth, C and M will denote a customer's account or a merchant's (account) name, respectively. S_C and S_M will denote BARTER realm servers maintaining C's and M's account, respectively. We assume the existence of a cryptographically strong hash algorithm (e.g., MD5 [28] or SHA [24]) and a safe

symmetric cypher (any algorithm could be used here, e.g., DES). A *hash* of an item will refer to a cryptographically strong hash function applied on the item. We further assume that messages sent between entities in the system are encrypted with the appropriate shared keys: Each pair of BARTER realm servers uses inter-realm keys for this matter; Users employ a session key established with the appropriate BARTER server.

4.1 Negotiation and Commitment

The basic idea in BARTER is to perform the item exchange by the intervention of BARTER realm servers. Initially, the parties negotiate a transaction by exchanging the relevant details: their account names, BARTER realm servers, a transaction identifier (see Section 4.2), possibly a price, and a human readable or a fixed format string describing the nature of the transaction, which is agreed upon in order to resolve potential content complaints. To allow both parties to work on common grounds, we introduce the notion of a *CTC* (Common Transaction Context), which is a record containing all the details regarding a single transaction to be processed; Both parties include all the relevant details in a *CTC*.

Negotiation is not restricted to a specific protocol. Note that BARTER does not provide a secure communication channel between the negotiating parties. Should privacy be a major concern, a secure channel may be established between the parties using one of the well-known methods (e.g., using SSL [3]). BARTER also supports pseudonyms, the use of which makes personal information difficult to collect.

During the negotiation phase, the parties send each other the items to be exchanged, encrypted with a randomly-chosen one-time symmetric key K, and a hash of this key. The keys used for encryption are delivered only *after* the transaction has been settled. In addition, both parties include a hash of the key they received and a hash of the encrypted goods in the *CTC*, before it is submitted for processing. The advantage of this approach (for a similar approach, see [16]) is threefold: it avoids sending electronic items through BARTER, guarantees delivery atomicity, and allows for content complaints to be resolved, as will be shown in Section 5.

At this point, the parties enter the commitment phase, and perform the following, depending on the type of protocol used: In the **LWF** protocol, both parties send the *CTC*s to their realm servers for processing. In the **RCV** protocol, the parties cross-verify commitments, by producing a digital signature of the *CTC*, and sending it to the other party for verification. Then, both parties include *their* signature and the other party's *verified* signature in the *CTC*, and send the *CTC* to BARTER for processing. The **AS-RCV** protocol incurs an asymmetric message flow, as shown in figure 2.

1. *M sends C his signature over the CTC.*
2. *C verifies M's signature and stores it. C sends M his signature over the CTC, and sends* BARTER *his CTC (including C's signature and M's* verified *signature).*
3. *M verifies C's CTC signature and stores it. M sends* BARTER *his CTC (including M's signature and C's* verified *signature);*

Fig. 2. Commitment in the **AS-RCV** Protocol

4.2 Transaction Processing

In the processing phase, BARTER processes the *CTC*s submitted at the commitment phase. In case the parties maintain their account in the same realm, their *CTC*s will be received by a single BARTER server which will process the transaction locally. If this is not the case, however, the corresponding BARTER servers must coordinate the transaction. Bearing in mind our assumptions regarding the network, environment and failure model, care must be taken not to leave the transaction in an inconsistent state. Below we sketch several issues involved in the transaction processing flow.

Atomic processing of a transaction. S_C and S_M settle the transaction using classic transaction processing techniques, namely, a protocol that resembles the Two Phase Commit (2PC) protocol [7]. S_C forwards C's *CTC* to S_M, which maintains two cached queues of pending transactions for merchants and customers. When a pair of *CTC*s relating to a specific transaction arrive at S_M, a local decision whether to execute or reject the transaction can be reached. Upon reaching a decision, S_M records it in stable storage and ensures that this decision makes its way through to S_C by collecting an acknowledgment from the latter. In case the transaction involves crediting or deducting the relevant account balances, S_C and S_M update their database accordingly. When evidence collection is required, both S_C and S_M record the relevant *CTC*s and the decisions reached in long term storage.

Execution/rejection decision. A transaction can be rejected by S_M for two possible reasons, the first of which could be lack of agreement between the parties involved in the transaction, and the second of which could be an attempt to re-authorize a previously processed transaction. To ensure that an agreement indeed holds between the parties, S_M must perform one of the following actions. In the **LWE** protocol, S_M must verify that both *CTC*s received are *field-wise identical.* In the two other protocol versions, **RCV** and **AS-RCV**, S_M must ensure that the cross-verified signatures submitted by both parties match (S_M does this simply by comparing the relevant strings).

Replay detection. Ensuring that transaction re-authorization does not occur (as a consequence of a commitment replay attempt) is achieved in BARTER using distributed bookkeeping and garbage-collection of globally-unique transaction

identifiers. Transaction identifiers are pairs (M, T), where M is a merchant name and T belongs to a set of serial numbers for merchant M. Each BARTER realm server maintains for each merchant M that holds an account in the realm, a list of M's processed (executed or rejected) transactions. This list is garbage collected by using a 'low watermark' method or by adding a coarse time dimension to the transaction identifier. Fresh transaction identifiers may be unjustly rejected in rare cases, but in no case is a stale transaction identifier accepted.

Transaction results, receipts and status queries. The transaction result messages (Figure 1, messages 6 and 7), provide users with notifications regarding the state of a transaction submitted by them for processing. In addition to these messages, a *signed receipt* generation process is carried in the background. This process requires passing through the entries in BARTER's long term storage, generating a signature for each entry, storing it and sending it to the appropriate users. These signatures can later be presented as transaction evidence to a third party or an arbitrator, as will be explained in the next section. The BARTER protocols also support *status queries*, which enable the users to query their BARTER server for the status of a transaction submitted for processing. Note that since the processing of a transaction usually involves several sites, and failures might occur, the status query protocol must take into account the fact that a transaction might be still in process, or failed processing.

5 BARTER Analysis

The BARTER system may encounter a wide variety of user conflicts and complaints, to which it must be able to respond. In case an arbitration process is required, BARTER must or supply the appropriate evidence. The protocols we suggested are robust enough to enable the resolution of most kinds of complaints and disputes.

Most complaints cannot be handled by means of an automated process. We assume the existence of widely accepted, trustworthy *arbitration* authorities, the decisions of whom can be legally enforced. These authorities could possibly be some sort of commercial network service. Complying to the terms of an arbitration authority would benefit a merchant's reputation.

When a dispute occurs, parties will obtain evidence, and file their complaints to the appropriate authority. The exact form of evidence, where they are obtained from, and their level of assurance is dependent upon the protocol used. Most disputes regarding a specific transaction will require both parties' CTCs. The level of assurance of evidence can be the BARTER's signature over the CTCs, or both parties' signature over the CTCs. Evidence can be stored by BARTER and obtained there, or retained by the parties.

To see how a content complaint can be resolved, assume one of the parties is provided with an encryption key K which decrypts a previously exchanged item, and is not satisfied with the item's nature or quality. This party can file a complaint to an arbitration authority, containing the appropriate evidence,

a description of the complaint and the appropriate encryption key. Complaints are examined by the arbitrator, who evaluates the item in question, and decides if it looks "reasonable". Of course, this arbitration process involves human consideration and judgment; in certain cases it can be automated (for example, determining if a digital item has a specific format). A possible decision of an arbitrator could be an instruction to refund a customer's account.

In the rest of this section, we briefly analyze some of the properties of the three suggested protocol versions.

The Light Weight Exchange Protocol: The **LWE** protocol does not require the parties to generate nor verify commitments. Due to this fact, denial complaints are unresolvable. The protocol is highly efficient, however, and may be used to support microtransactions. Content complaints are resolvable: BARTER can provide the users with a signed receipt regarding the transaction in question, which can be filed as evidence to an arbitration authority.

LWE satisfies delivery atomicity. Both parties include their one-time encryption keys in the *CTC*s they submit to BARTER. These keys are verified to be the pre-image of their claimed hash, and forwarded to the relevant parties only if the transaction can be processed (i.e., an agreement over the details holds, and no replay attempt is made). Moreover, these keys will eventually reach their destination, even in the presence of communication failures or network partitions, since they are recorded in long term storage as part of the *CTC*.

Note that the commitment point for both parties in this protocol is sending the encrypted *CTC* to BARTER for processing.

The Remote Commitment Verification Protocol: According to the **RCV** protocol, parties must cross-verify *CTC* signatures. Since BARTER does not perform signature verification, it would seem that it can be driven to process a transaction although it does not have ample evidence for the parties' commitment to proceed. Fortunately, this is not the case. By ensuring that two pairs of identical signatures are submitted, BARTER can always be assured of their authenticity. Of course, if both parties collude, this is not the case. It is quite doubtful, however, that the parties would have an incentive to cooperate in deceiving BARTER.

Consequently, denial complaints are resolvable, since BARTER can provide both parties' commitments for each previously processed transaction. Transactions can not be forged by BARTER, since each and every transaction must be upheld by both parties' commitments. As in the **LWE** protocol, content complaints are resolvable, only this time, the evidence submitted to an arbitration authority is the parties' commitments. Similarly, delivery atomicity is achieved.

The commitment point in this protocol is sending the *CTC* to BARTER. Note that sending two different signatures to BARTER and to the other party may potentially cause an unresolvable dispute. Should this be the case, the commitment stored by BARTER is the one that counts.

The Asymmetric Remote Commitment Verification Protocol: Note that the special case in which digital items are traded for payment, gives the merchant a

certain advantage over the customer. The encryption key can be forwarded to the customer after the payment has been cleared, or even after the customer has given his consent to complete the transaction. This implies that online intervention of a trusted party is not required in this case, and the parties may exchange, verify and store commitments through an asymmetric message flow, as demonstrated in Figure 2. This message flow ensures that the merchant gets a hold of the customer commitment and verifies it, before forwarding the decryption key to the customer (otherwise the customer can deny consenting to the transaction, get his money back and nevertheless enjoy the item delivered). Furthermore, to circumvent a similar denial by the merchant, the customer does not send his commitment to the merchant unless he has already received, verified and stored the merchant's commitment. Denial and content complaints are resolvable in this protocol, however the parties themselves must provide the appropriate evidence.

The commitment point in this protocol is the point by which a party sends his signed CTC to the other party for verification (see the discussion above).

6 Conclusions and Future Directions

We have presented BARTER, a credit-debit electronic commerce system intended to be deployed over open, large-scale, global and heterogeneous networks.

We have shown how BARTER deals with aspects of system behavior that are likely to suffer service degradation unless potential bottleneck points are distributed or replicated. These aspects are the system servers load, the authentication subsystem, the account management subsystem, and the maintenance of global data (such as transaction identifier lists). In addressing these issues, BARTER takes into account the inherent asynchronous, unreliable, insecure and failure-prone environment assumptions.

Note that BARTER servers are still single points of failure. Further research could thus focus upon replicating realm servers, which would allow transparency of service without inviting new classes of attacks, or upon supporting some kind of realm overlapping. An example of a consistent object replication layer built over the Transis communication subsystem can be found in [17].

BARTER supports simultaneous item delivery, however no timely delivery properties are yet assured. We are investigating ways to add temporal constraints to item delivery, by ensuring a continuous interaction with a BARTER server. This notion can be used for quality-assured delivery of large, continuous objects such as video streams.

BARTER's multi-party exchange service can be used as a basic building block for extended services, such as online auctions. Defining requirements for such extensions and their efficient protocol support remain a suggestion for future research.

References

1. First Virtual Holdings Inc. URL: `http://www.fv.com`.
2. Mondex International Ltd. URL: `http://www.mondex.com`.
3. P. K. Alan O. Freier and P. C. Kocher. Secure Sockets Layer 3.0, Mar. 1996. Internet Draft.
4. N. Asokan, M. Schunter, and M. Waidner. Optimistic protocols for fair exchange. In *Fourth ACM Conference on Computer and Communications Security*, 1997.
5. N. Asokan, V. Shoup, and M. Waidner. Asynchronous protocols for optimistic fair exchange, 1997. IBM Zurich Research Report, RZ 2976 (#93022).
6. M. Bellare, J. Garay, R. Hauser, A. Herzberg, H. Krawczyk, M. Steiner, G. Tsudik, and M. Waidner. iKP – A family of secure electronic payment protocols. In *First USENIX Workshop on Electronic Commerce*, pages 89–106, New York, July 1995.
7. P. Bernstein, V. Hadzilacos, and N. Goodman. *Concurrency Control and Recovery in Database Systems*, chapter Distributed Recovery, pages 217–264. Addison-Wesley, 1987.
8. J. P. Boly, A. Bosselaers, R. Cramer, R. Michelsen, S. Mjølsnes, F. Muller, T. Pedersen, B. Pfitzmann, P. de Rooij, B. Schoenmakers, M. Schunter, L. Vallée, and M. Waidner. The ESPRIT project CAFE - high security digital payment systems. In *Proceedings of the Third ESORICS*, Nov. 1994. Lecture Notes in Computer Science No. 875.
9. S. Brands. Untraceable off-line cash in wallets with observers. In *Advances in Cryptology: Crypto 93*, 1994.
10. M. Burrows, M. Abadi, and R. Needham. A logic of authentication. *Proceedings of the Twelfth ACM Symposium on Operating Systems Principles*, 23(5), Dec. 1989.
11. L. J. Camp. *Privacy and Reliability in Internet Commerce*. PhD thesis, Carnegie Mellon University, School of Computer Science, Aug. 1996. Technical Report CMU-CS-96-198.
12. L. J. Camp, M. Sirbu, and J. D. Tygar. Token and notational money in electronic commerce. In *Proceedings of the first USENIX Workshop on Electronic Commerce*, New York, July 1995.
13. D. Chaum. Security without identification: transaction systems to make the big brother obsolete. *CACM*, 28(10):1030–1044, Oct. 1985.
14. D. Chaum. Achieving electronic privacy. *Scientific American [International Edition]*, 267(2):76–81, Aug. 1992.
15. D. Chaum, A. Fiat, and M. Naor. Untraceable electronic cash. In *Proc. CRYPTO 88*, pages 319–327. Springer-Verlag, 1988. Lecture Notes in Computer Science No. 403.
16. B. Cox, J. D. Tygar, and M. Sirbu. NetBill security and transaction protocol. In *Proceedings of the First USENIX Workshop on Electronic Commerce*, New York, July 1995.
17. I. Keidar and D. Dolev. Efficient Message Ordering in Dynamic Networks. In *ACM Symp. on Prin. of Distributed Computing (PODC)*, number 15, May 1996.
18. J. Kohl and B. Neuman. Internet Request for Comments: The Kerberos Network Authentication Service (V5), 1993. RFC 1510.
19. J. T. Kohl, B. C. Neuman, and T. Y. Ts'o. The evolution of the Kerberos authentication service. In *Proceedings of the Spring 1991 EurOpen Conference*, 1991.
20. L. Lamport, R. Shostak, and M. Pease. The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, July 1982.

21. M. S. Manasse. The Millicent protocols for electronic commerce. In *Proceedings of the first USENIX Workshop of Electronic Commerce: July 11–12, 1995, New York, New York, USA*, 1995.
22. MasterCard and Visa. *Secure Electronic Transaction Specification*, draft edition, June 1996. URL: `http://www.mastercard.com/set/set.htm`.
23. G. Medvinsky and B. C. Neuman. NetCash: A design for practical electronic currency on the internet. In *Proceedings of 1st the ACM Conference on Computer and Communication Security*, Nov. 1993.
24. National Institue of Standards and Technonology (NIST). *FIPS Publication 180: Federal Information Processing Standard: Secure Hash Standard (SHS)*, May 1993.
25. National Institue of Standards and Technonology (NIST). *FIPS Publication 186: Federal Information Processing Standard: Digital Signature Standard (DSS)*, May 1994.
26. B. C. Neuman and G. Medvinsky. Requirements for network payment: The NetCheque perspective. In *Proceedings of IEEE COMPCON*, 1995.
27. T. P. Pedersen. Electronic payments of small amounts. In *Cambridge Workshop on Security Protocols*, 1996.
28. R. L. Rivest. Internet Request for Comments: The MD5 Message-Digest Algorithm, Apr. 1992. RFC 1321.
29. R. L. Rivest and A. Shamir. Payword and Micromint: Two simple micropayment schemes. Technical report, MIT LCS, 1995.
30. R. L. Rivest, A. Shamir, and L. Adelman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2), Feb. 1978.
31. G. Shamir. A scalable and efficient electronic commerce system. Master's thesis, Institute of Computer Science, Hebrew University of Jerusalem, Oct. 1997.
32. J. Su and J. D. Tygar. Building blocks for atomicity in electronic commerce. In *6th USENIX Security Symposium, San Jose, CA*, Berkeley, CA, USA, July 1996.
33. L. Tang. Verifiable transaction atomicity for electronic payment protocols. In *ICDCS '96: Proceedings of the 16th International Conference on Distributed Computing Systems, Hong Kong*, 1996.
34. B. Tung, C. Neuman, J. Wray, A. Medvinsky, M. Hur, and J. Trostle. Public key cryptography for initial authentication in Kerberos, Oct. 1995. Internet Draft.
35. J. D. Tygar. Atomicity in electronic commerce. In *PODC: 15th ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing*, 1996.
36. S. von Solms and D. Naccache. On blind signatures and perfect crimes. *Computers and Security*, 11(6):581–583, Oct. 1992.