The following paper was originally published in the
Proceedings of the Sixth USENIX UNIX Security Symposium
San Jose, California, July 1996.

Chrg-http: A Tool for Micropayments on the World Wide Web

Lei Tang, Carnegie Mellon University
Steven Low, University of Melbourne

For more information about USENIX Association contact:

1. Phone: 510 528-8649
2. FAX: 510 548-5738
3. Email: office@usenix.org
4. WWW URL: http://www.usenix.org

Chrg-http: A Tool for Micropayments on the World Wide Web

Lei Tang * Steven Low †

Abstract

Chrg-http is a simple and secure protocol for electronic payments over the Internet, especially in an intranet environment. It is designed to support those micropayments (or more specific, electronic publishing, which have costs ranging from pennies to a few dollars. A widely used secure system Kerberos V5 has been incorporated into the http protocol. The security and authentication of a transactions is provided by Kerberos, without expensive public key cryptographic computations, or on-line processing through a centralized payment processing server, which is the case to most of the existing electronic payment systems on the World Wide Web. Our implementation is based on the billing model (or the subscription model). The simplicity of the model also helps to reduce the charging cost overhead.

1 Introduction

The World Wide Web (WWW) [4] has become a very popular facility for dissemination of information on the Internet. However, most of the information on the Internet is free. Intellectual property owners have little incentive to make valuable information accessible through the computer network. An electronic payment mechanism should be provided to facilitate monetary transactions among the customers and the potential merchants on the Internet. There exists several proposed payment systems based on the World Wide Web [3, 1, 2]. However, expensive public key cryptographic algorithms are widely used for the purpose of authentication and security. While these payment systems are ideal for those high-value transactions, they are not suitable for micropayments, in which most of the information goods have values ranging from pennies to a few dollars.

In this paper, we describe the design and implementation of a lightweight electronic payment protocol called the *chrg-http* protocol. It is intended for electronically publishing copyrighted goods such as journal papers or digital images on the Internet [6]. The protocol is based on the simplest charging model–the billing model (or the subscription model). Authentication and audit are two key issues in this model. A widely used authentication system Kerberos Version 5 [9] has been incorporated into the Cern httpd server and the NCSA Mosaic browser to provide authentication and secure communication between the browser and the httpd server. The kerberized http protocol is the chrg-http protocol. The chrg-http protocol is implemented by adding a new method called CGET into the http protocol. The kerberized Mosaic browser supports a new URL access method called "chrg-http", for connecting to the kerberized httpd server.

On the httpd server side, we add a module called *CGET handler* into the Cern httpd daemon. The CGET handler processes the CGET requests from the Mosaic browser, authenticates the customer's identity, encrypts the file requested by the customer, and sends it to the Mosaic browser. The CGET handler also logs access information for generating the bills which are sent to the customers at regular intervals.

On the client side, we add a module called the *chrg-http handler* into the NCSA Mosaic browser. The chrg-http handler translates a chrg-http request into a request including CGET as its method which is recognizable to the httpd server. It also authenticates the customer's identity to the CGET handler, decrypts the encrypted file, and passes the file to the browser for display.

Our presentation is organized as follows. First, we describe the project goals, the design decisions, and the reasons behind them. Second, we give a brief introduction to the Kerberos V5 system, we also describe how we incorporate the Kerberos authentica-

*GSIA, Carnegie Mellon University. Pittsburgh, PA 15213-3891. Email: ltang@cmu.edu. The work was done when the author was a summer intern in AT&T Bell Laboratories in the summer of 1994. The views and conclusions contained in this document are solely those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of Carnegie Mellon University, or AT&T Bell Laboratories.

†Dept. of Electrical and Electronic Engineering, Univ of Melbourne, Victoria 3052, Australia. Formerly a member of technical staff in AT&T Bell Laboratories.

tion protocol into the http protocol. Then we describe our implementation of the chrg-http protocol on the World Wide Web. We will show how the protocol is incorporated into the NCSA Mosaic browser and the Cern httpd server.

2 Goals

The goal of our system is to allow micropayments such as electronic publishing through the World Wide Web, especially in an intranet environment. We try to demonstrate that the systems built solely for the purpose of authentication in distributed systems could be extended to handle electronic payments in a distributed environment, especially for those micropayments. Our goal is to reduce the charging cost overhead by adopting a simple billing structure and using the existing distributed authentication infrastructure for payment security and nonrepudiation. Any dispute regarding a transaction between a customer and a merchant could be arbitrated by the trusted party, the key distribution center of the authentication system.

Security
We use security to refer to the most important properties of an system for electronic payment: secrecy, authenticity, integrity, and nonrepudiation. Secrecy refers to denial of access to information by unauthorized principals (e.g., eavesdropper on the network). Authenticity refers to validating the source of a message; that is, the message was transmitted by a properly identified sender and is not a replay of a previously transmitted message. Integrity refers to assurance that a message was not modified accidentally or deliberately in transmit by replacement, insertion, or deletion. Since our system is based on the billing model, authenticity is our main concern.

Low costs
Our system is intended for electronic publishing. Most of the items for sale have cost ranging from pennies to several dollars. We design our protocol as cheap as possible so that the transaction cost is a small fraction of the cost of the purchased item. Our system reduces the charging overhead in a number of ways:

- We use a simple charging model (the billing model) to build our system. The simple accounting structure lowers the cost of development and maintenance, hence the charging overhead.

- The authentication mechanism provided by Kerberos is strong enough so that the cost of breaking the system is far more higher than that of purchasing the items directly from the merchant. Moreover, Kerberos provides a lightweigh authentication mechanism based on private key cryptographic algorithms (e.g. DES). It is much efficient compared with those protocols based on the public key cryptographic algorithms, such as the EIT Secure-http protocol and the Netscape HTTPS protocol. This reduces the computational overhead, and the corresponding public key certification and management cost, hence reduces the charging overhead.

- Kerberos are widely used by many institutions for system and network security. The kerberized browser and httpd server can be incorporated into a local environment smoothly. This reduces the administrational overhead. This also makes our system an ideal candidate for micropayments in an intranet environment.

Transparency
We incorporate our chrg-http protocol with the Mosaic browser. The charging mechanisms inside the protocol should be transparent to the customers.

Compatibility
Our implementation of the security enhanced World Wide Web should be compatible with the existing httpd servers and Mosaic browsers so that the customers who use our browser can still access those old httpd servers and the customers who access our httpd server using the old Mosaic browser can still get those files which are free from our httpd server. Instead of assigning a new port number to the new protocol, we use the same port number used by the http protocol.

3 Design of the chrg-http protocol

3.1 Charging models

We have to select a charging model before we design our protocol. There are several commonly used models for designing our protocols. They are the billing model (or the subscription model), the credit card model, the electronic check model, and the debit model.

In the credit card model, the customer sends his credit card number to the merchant through some pre-established secure channel between them. We did not use this model because the credit card transaction cost is high (standard charging costs levied by the credit card company). It is not suitable for

our system, which is intended for the low cost transactions such as electronic publishing of journal papers. The SEPP (Secure Electronic Payment) protocol proposed by IBM, Mastercard, and Netscape is such an example [1].

The debit model is a good candidate for low-cost transactions. In this model, a set of electronic payment service providers are established to transfer funds between the customer's account and the merchant's account. We do not adopt this model for complexity reason. In this model, not only does the payment service provider need to provide authentication mechanisms between the customer and the merchant, but it also needs to provide financial services (e.g., funds transfer, fund management, etc) for the customers and the merchants. Since our project should be finished in three months and building a central billing server from scratch is not an easy task, we did not use this model either. But we did design a set of "cheap" protocols based on this model described in [10].

In the electronic check model, the customer signs digital checks against his checking account. The recipient of the digital check could verify the validity of those checks through the public key certificate issued to the signer by a trusted party. The electronic check has the best scalability among all these models. However, a scalable public key management system is very difficult to implement. So it is not an ideal candidate for our system. The USC NetCheck is such an example [8, 7].

We use the billing model for simplicity reason. In the billing model, every customer registers with different merchants and obtains services (or goods) from the merchants. Whenever a customer and a merchant want to begin a transaction, they authenticate with each other. If the authentication succeeds, then the merchant logs the customer's identity and purchases items, sends a bill to the customer at regular intervals. Authentication and audit are two key issues in this model. Authentication guarantees that only the subscribed members of the merchant can access this service. Audit guarantees the preciseness of the bill sent to his customers by the merchant. We have implemented our chrg-http protocol based on this model. The implementation details are deferred to the next section.

3.2 The incorporation of http and Kerberos V5

Our chrg-http protocol is a combination of the http protocol and the Kerberos authentication protocol.

The Kerberos authentication system is a security system and has been adopted by many organizations as a solution to the network authentication problem. Kerberos provides evidences of a principal's identity. A principal is generally either a user or a particular service on some machine. A principal consists of three parts: the name of a user or a service, the instance either null or representing particular attributes of the user, and the realm representing the domain the user belonging to.

Kerberos principals obtains tickets for services from a special server known as the ticket-granting server (or key distribution center). A ticket contains assorted information identifying the principal, encrypted in the private key of the service. We briefly explain the chrg-http protocol as follows. The details of the Kerberos protocol V5 can be found in [9].

The customer sends the identity of the merchant to the Kerberos Key Distribution Center (called KDC) and then he is prompted for a password to prove his identity to the KDC. The KDC generates a ticket including the identity of the customer, the identity of the merchant, the customer's IP address, the timestamp, the expiration time, and a random session key K_{cm} which will later be used for secure communication between the customer and the merchant. This is all encrypted in a key K_{ma} known only to the KDC and the merchant. The KDC then sends the ticket, along with a copy of the session key K_{cm} and some additional information, back to the customer. The response is encrypted in the customer's private key K_{ca}, known only to the KDC and the customer. Once the customer receives the message from the KDC, he obtains and sends the ticket to the merchant. The merchant verifies the ticket and sends the encrypted file back. The merchant can also prove his identity to the customer using the mutual authentication mechanism provided by the Kerberos system.

Now we begin to describe how the Kerberos V5 protocol is incorporated with the http protocol [5]. We add a new method `CGET` into the method field of the http request structure. Besides the functions of obtaining a file for the Mosaic browser, `CGET` also completes a mutual authentication between the httpd server and the Mosaic browser using the protocol described above. We do not put the authentication tokens (Kerberos ticket) in the MIME header as suggested by the http protocol specification. If we did so, an adversary could launch an active attack on this scheme. The adversary could obtain the message header by eavesdropping the communication. Then he takes out the authentication tokens from the header and puts them into other messages, since there is no way to guarantee the integrity of the

message in the http protocol. Another reason is that the message format of the Kerberos ticket is not consistent with the specified http message format. Instead of putting the Kerberos ticket into the MIME header, we pass the socket connection between the browser and the httpd server to the Kerberos authentication routines. The httpd server and the Mosaic browser pass the Kerberos tickets through the TCP connection. After the authentication succeeds, the messages passed between the httpd server and the Mosaic browser are the same as the message format specified by the http protocol except that the content of the message is encrypted.

In order for the customer and the merchant to do monetary transaction through chrg-http, the merchant has to register a secret key in the customer's realm, or the customer has to register a secret key in the merchant's realm, or both the customer's realm and the merchant's realm has to share a secret and authenticate each other through the Kerberos cross-realm authentication mechanism.

Another inconsistency between the Kerberos protocol and the http protocol is the specification of the user identities. In the http protocol, the user identity is denoted by a Unix Id (i.e., a number). Two different users on two different operating systems may have the same user identity. In order to make sure that the identity of the user is unique, the Kerberos principal name should be used as the user's identity. This is another modification to the http protocol.

3.3 Comparison to other secure http protocols

The most related work to chrg-http is the Secure http protocol designed by EIT and the Secure Socket Layer (SSL) protocol designed by Netscape. The goal of both protocols is to extend the http protocol for secure communication and authentication although their approaches are slightly different. Public key cryptographic algorithms such as RSA, and public key based systems such as PEM and PKCS-7, are widely used to achieve secure communication and authentication. In order to run the secure http protocols, either a unique port number is assigned to accomplish the protocol in a different channel other than the channel used by the http protocol (SSL), or the specifications of HTML, URL, and http header format have to be modified (secure http). Although these protocols scale better than our chrg-http protocol, computationally they are more expensive than our protocol. Moreover, it is assumed that there exists a trusted public key certification center to certify every public key used in the protocols. The complexity of the public key management will increase the charging overhead of those systems.

The Netscape https protocol based on SSL is implemented on a separated communication channel. A unique TCP port number is assigned to the https protocol. We use the same communication channel used by the http protocol to implement our chrg-http protocol. We pass the socket connection between the browser and the httpd server to the Kerberos authentication routines. The additional communication channel is not necessary in our protocol.

4 Implementation

Our chrg-http protocol is implemented in C based on NCSA Mosaic version 2.4 and Cern httpd server version 1.0 plus Kerberos V beta 4.2. In this section, we first introduce the World Wide Web, give an overview of the components of the system. Then we describe how we incorporate our protocol with the Mosaic browser and the httpd server.

4.1 The World Wide Web

The fundamental mechanism underlying the Web is the HyperText Transfer Protocol (also called the HTTP protocol). The HTTP protocol is a stateless, object-oriented, and request/response protocol. A client establishes a connection with the server. The client sends a request to the server in the form of a request method, a universe resource locator(URL), and a protocol version, followed by a MIME-like message containing request modifiers, client information, and possible body content. The server responds with a status line (including its protocol version and a success or error code), followed by a MIME-like message containing server information and possible body content.

The Web server (here we mean Cern httpd server) and the Web client (here we mean the NCSA Mosaic browser) use the HTTP protocol described above to exchange information. Here is the general scenario of the Web currently in use: The Web server exports a hierarchical file space to the Web clients. The file space consists of documents of various types. The primary function of the Web clients is to acquire documents from this file space in response to user actions. The client and the server must also negotiate a document format which is acceptable to them both. The Web servers and clients communicate with each other over TCP/IP streams.

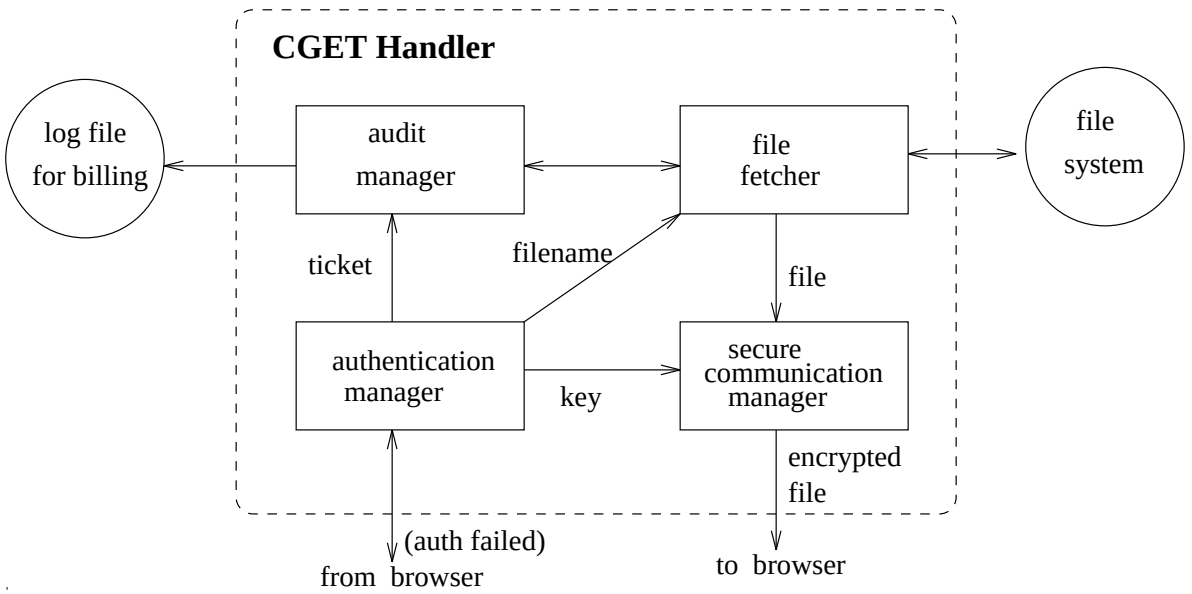

Figure 1: The structure of the CGET handler

4.2 The httpd server

We implement an independent module called the *CGET handler* and plug it into the Cern httpd server (version 1.0). The structure of the CGET handler is shown in Figure 1.

The CGET handler consists of four parts: the *authentication manager*, the *secure communication manager*, the *audit manager*, and the *file fetcher*. The authentication manager implements the Kerberos mutual authentication protocol between the Mosaic browser and the httpd server. If the authentication fails, the authentication manager sends an error message back to the Mosaic browser. If the authentication succeeds, the authentication manager caches the session key which will be used for secure communication between the httpd server and the Mosaic browser. The authentication manager also passes the name of the file to the file fetcher and the authentication token (ticket) to the audit manager. If the file fetcher finds the file in the local file system, the file is fetched and is passed to the secure communication manager. The secure communication manager encrypts the file using the cached session key and sends the encrypted file to the Mosaic browser. The way that the session key being cached is exactly the same as the way in Kerberos. In the mean time, the audit manager logs the following information to the log file: the customer identity, the IP address of the browser, the file name, the price, the timestamp and the authentication token. The log file will be used to generate the billing information for the customer and to handle possible dispute between the customer and the merchant.

In order to make sure that our new protocol is compatible with those web browsers and httpd servers in which our protocol is not implemented, we employ a new indirection method. Every copy of nonfree file is associated with a tag file in the same directory of the local file system or in some specified repository. The tag file serves two functions here: one is to serve as an access control list; another one is to provide price information and advertisement information. We illustrate our method with an example.

When a customer clicks on a hyperlink anchored by `http://www.cmu.edu/paper.ps`, the client opens a TCP connection with the httpd server on the machine `www.cmu.edu` and sends request `GET paper.ps` to the server. The server then checks if the tag file `paper.ps_chrg` exists or not. If the file exists, then the file `paper.ps_chrg`, together with an error message "402 Payment Request", are sent to the client. This file contains the price and advertisement information of the file `paper.ps` and a URL link `chrg-http://www.cmu.edu/paper.ps` to the file `paper.ps`. If the customer decides to purchase the document `paper.ps`, he clicks on the chrg-http URL link. The client translates the chrg-http URL link into the request `CGET paper.ps` and sends the request to the server. In the mean time, the customer is asked by a pop-up window to input his authentication information and the price of the item. The customer is also asked if the file `paper.ps` needs to be encrypted when the server sends the file to the

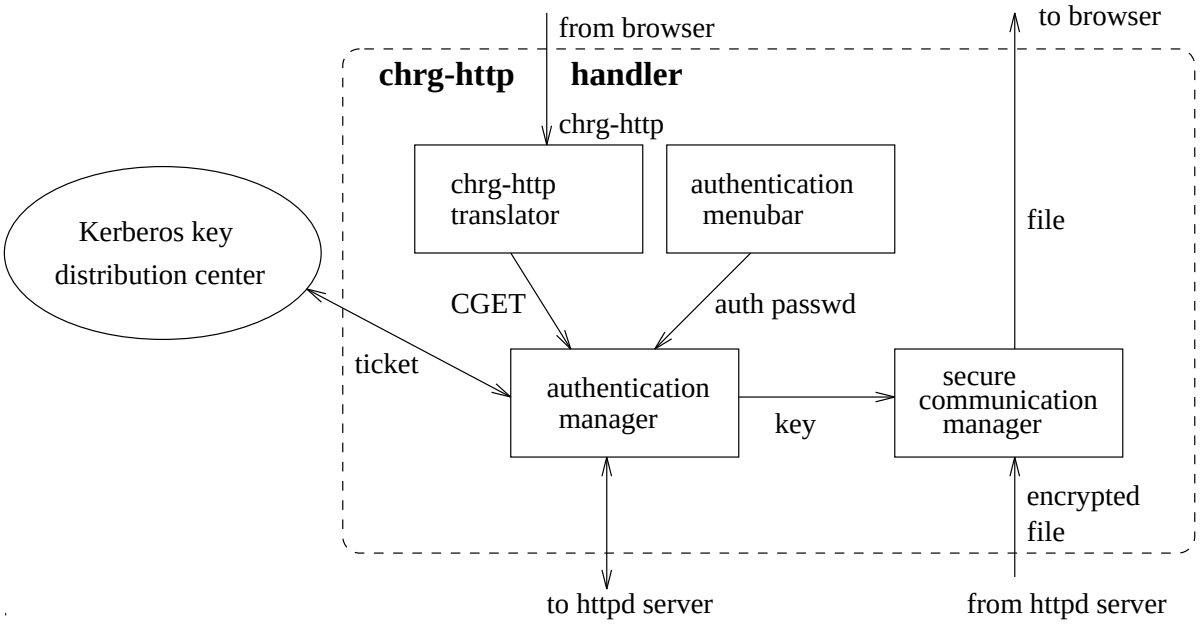

Figure 2: The structure of the chrg-http handler

client. When the server receives the request `CGET paper.ps`, it passes the request to the CGET handler.

One subtlety is how to handle the communication failure occurred during the execution of the chrg-http protocol. Since the http protocol is stateless, the httpd server does not know if the Mosaic browser receives the item it sends. If the CGET handler logs after the item is sent out from the server, the information may get lost due to network partitions. The customer does not receive the purchased item but his account is updated. This is unacceptable to the customer. We can not solve this problem in our chrg-http protocol since the Mosaic browser does not acknowledge the httpd server if it had received the items or not. This problem can only be resolved manually at the present time. If a customer complains that he has not received a purchased item, The administer of the server checks the log file and sends the item to the customer directly.

In case that the encryption of the purchased item is not desirable due to either political reasons (e.g., US government export regulation), or other technical reasons (e.g., the file includes multimedia data and the amount of data is huge, encryption of the data will degrade the performance of the server dramatically), we introduce a method called *one-time URL* to enhance the security of our protocol so that only those customers who have authenticated their identities to the server can get the purchased items. We illustrate our method with an example. Suppose that a customer wants to purchase a copy of video by clicking on a URL `chrg-http://www.cmu.edu/Jackson.mpeg` (the size of the file `Jackson.mpeg` is very big). If the customer passes the authentication, instead of encrypting the file, the server invents a random string R. The value of R is unpredictable. The server makes a symbolic link `R.mpeg` to `Jackson.mpeg`. Then the server sends URL `http://www.cmu.edu/R.mpeg` to the customer through the secure channel established by Kerberos and the customer can obtain the file from the server. Once the file `R.mpeg` is accessed, a daemon process (which is a part of the file handler) running on the server side unlinks the symbolic link.

4.3 The Web browser

Our implementation of the Web browser is based on the NCSA Mosaic version 2.4. We implement a module called the *chrg-http handler* and plug it into the Mosaic browser. The structure of the chrg-http handler is shown in Figure 2.

The chrg-http handler consists of four parts: the *authentication manager*, the *secure communication manager*, the *chrg-http translator*, and the *authentication menubar*. The chrg-http translator translates a
URL such as `chrg-http://www.cmu.edu/paper.ps` into `CGET paper.ps` and passes `CGET paper.ps` to the protocol handler inside the Mosaic browser. The protocol handler invokes the authentication man-

ager to authenticate mutually with the httpd server `www.cmu.edu`. The authentication manager selects the authentication method (right now the only choice is Kerberos V) through the authentication menubar and inputs the authentication password. Then the authentication manager obtains a ticket from the Kerberos key distribution center (KDC). The authentication manager caches a session key into the local file system (accessible only by the browser itself) used for decrypting files from the httpd server. It also implements the mutual authentication protocol with the httpd server based on the Kerberos ticket issued by KDC. If the authentication succeeds, the httpd server will send the encrypted file to the secure communication manager. The secure communication manager decrypts it and passes it to the browser and the procedure completes.

5 Conclusion

We have incorporated Kerberos V5 with the Cern-httpd server and NCSA Mosaic to provide a tool for electronic publication on the World Wide Web. This work was done in the summer of 1994. Since it is a three-month project, we do not have performance data to evaluate the efficiency of our system.

6 Acknowledgements

We are grateful to David Kristol and Nicholas Maxemchuck for their helps and discussions while this system was being designed and implemented. The first author was supported by a summer internship from AT&T Bell Labs.

References

[1] Secure Electronic Payment Protocol Specification. http://www.mastercard.com/Sepp/sepptoc.html, 1995.

[2] The Secure HyperText Transfer Protocol. http://www.eit.com/creations/s-http/draft-ietf-wts-shttp-00.txt, 1995.

[3] The SSL protocol. http://www.netscape.com/newsref/std/SSL.html, 1995.

[4] T. Berners-Lee, R. Cailliau, A. Luotonen, and H. Nielsen. The World-Wide Web. *Communication of the ACM*, August 1994.

[5] T. Berners-Lee, R. Fielding, and H. Frystyk Nielsen. Hypertext Transfer Protocol–HTTP/1.0. Internet Draft, March 1995.

[6] N. Maxemchuk. Electronic Document Distribution. *AT&T Technical Journal*, pages 73–80, September 1994.

[7] C. Neuman. Proxy-based authorization and accounting for distributed systems. In *Proceedings of the 13th International Conference on Distributed Computing Systems*, pages 283–291, May 1993.

[8] C. Neuman and G. Medvinsky. Requirements for Network Payment: The Netcheque Perspective. In *Proceedings of IEEE Compcon'95*, March 1995.

[9] C. Neuman and T. Ts'o. Kerberos: An Authentication Service for Computer Networks. *IEEE Communications*, 32(9), September 1994.

[10] Lei Tang. A Set of Protocols for Micropayments in Distributed Systems. In *Proceedings of First USENIX Workshop on Electronic Commerce*, July 1995.