Modeling an Electronic Commerce Protocol for a Trusted Banking

By

Sahab Dhiaa Mohammed

Ministry of Higher Education and Scientific Research / Iraqi Commission for Computers and Informatics

Abstract:

E-commerce Protocols represent the allowed interactions among communicating components. Protocols are essential in electronic commerce to constrain the behaviors of autonomous entities. This paper proposed an e-commerce protocol that consists of two players: merchant and customer. The two e-commerce parties i.e., merchant and customer perhaps lie in far distance. The merchant has its bank and the customer also has its bank and the two banks (merchant bank and customer bank) are connected to a trusted bank. The trusted bank is an international government bank. The function of the trusted bank is to check the validity of the customer credit card and carry out the required money transfer. The bank protocol can detect duplicated, reused, or expired credit card. The proposed protocol guarantees fairness. That is by engaging in the protocol, no one in the e-commerce transaction i.e., merchant or customer, can gain advantages over the other players by misbehaving, misrepresenting or by prematurely aborting the protocol. The merchant site has been built as a web service model. This model is based on open standards and protocols: HTTP and XML-based protocols including SOAP (Simple Object Access Protocol), WSDL (Web Service Description Language, and UDDI (Universal Description Discovery and Integration).

Keywords: authenticated bank; fairness; web service; open standards; SOAP; WSDL; UDDI

1. Introduction:

The growing popularity of the World Wide Web has resulted in an increased interest in e-commerce. Consequently, a number of e-commerce protocols have been proposed. Most of these protocols must ensure that the information exchanged the parties involved in the e-commerce, is protected from unauthorized disclosure and modification. In addition, payment transaction supplies advanced services for merchants, customers and trusted third party plays an important role in the implementation of e-commerce system. Therefore, the issue of confidentiality and safety for the electronic payment protocol over Internet becomes popular topics [1][2][3]. This report often needs to develop the electronic payment system over Internet in e-commerce applications. Formal method can be used to analyze a payment protocol so that we can make the design of e-commerce system trustworthiness. This paper shows how to model a typical electronic payment protocol; the transaction among the three parties, i.e., the merchant, the customer, and the bank that conducted to the merchant involves for payment process and had been selected by the customer. In today’s Internet-driven society, most financial institutions offer their customers money transfers and other financial services (from account consultation to account aggregation, bills payment, international funds transfer, etc.) through the Internet channel. Online banking has quickly become an integral part of financial institutions’ overall strategy. The number of online-banking users has tremendously increased and should keep on increasing in the years to come. By accessing banking services from any place and at any moment, end-users can benefit from increased convenience, simplicity and fastness. Besides, banks can reduce their transaction costs as e-banking is five times* cheaper than traditional banking ways. They can strengthen their core business and broaden their customer scope by reaching valuable customers, selling new financial products and using e-services as an attractive differentiating tool [4].

2. XML-Related Issues

Two of the most important data handling technologies are XML documents and relational databases. While relational databases have been used to store a significant amount of today's business data, XML documents have rapidly gaining momentum in e-commerce and Internet-based information exchanges. Hence, an essential part of many XML-based applications is data exchange between relational databases and XML documents. A generic load/extract utility for data transfer between XML documents and relational databases." They identified and solved four specific problems [5]:

(1) Loading data from XML documents into relational tables with a known schema.

(2) Creating XML documents according to a known DTD from data extracted from a database.

(3) Generating relational schemas from XML DTDs for on-the-y storage of XML documents.

(4) Generating XML DTDs from relational schemas for on-the-y extraction of relational data.

The purpose of a Document Type Definition or DTD is to define the structure of a document encoded in XML. It is possible to build and use files containing XML tags without ever defining what tags are legal. Two definitions:

• A well-formed file is one that obeys the general XML rules for tags: tags must be properly nested, opening and closing tags must be balanced, and empty tags must end with '/>'.

• A valid file is not only well-formed, but it must also conform to a publicly available DTD that specifies which tags it uses, what attributes those tags can contain, and which tags can occur inside which other tags, among other properties [6].

The advantage of a valid file is that its contents are more predictable for applications that want to process or present that file. The DTD insures that only certain tags can be used in certain places. Although XML documents can be accompanied by a DTD that defines the structure of the documents, the presence of a DTD is not mandatory. The difficulty in deriving the DTDs for XML Documents lies in the fact that DTDs are of different syntax from XML documents and that prior knowledge of the structure of the documents is required. With DTD-Miner, a user can submit a set of similarly structured XML documents and the system will automatically suggest a DTD. It is flexible, scalable, and suitable for distributed network environments. The application was aimed at supporting data exchange in ecommerce systems.

3.New Roles of DB Management Systems

Databases provide a foundation for many types of information systems and can be thought of as the cornerstone of modern information technology. With the emergence of component-based software development, the role of database management systems may also change. The traditional view of information system development is to think of a database management system as the platform and the information systems developed as applications of databases. This view may prove to be limiting in component-based software engineering and new views might emerge that take the e-commerce applications as the principle system and databases as servant components. New concepts in future databases are evolving towards support for electronic commerce. For instance, evolving databases are suggested to support negotiations in electronic commerce. The term "evolving" stresses the extremely dynamic nature of a negotiation, which requires changing the product descriptions, orders, and even protocols of negotiation on the fly. Another example is supporting EDI via distributed databases, which support the exchange of EDI messages through database transactions. As a result, the application-to-application communication is replaced by database-to-database communication.

4. XML-based Document Databases

The document-centric interoperable e-commerce frameworks will rely heavily on database services to deposit and access the common business library and the domain specific information. These document databases will provide new challenges to database research since their requirements are different from the conventional business transaction databases. The data in XML-based documents are likely to be semi-structured and the transactions to the documents are mostly read-accesses. Although there are already several applications on the market that store XML documents in relational databases it is not clear if relational database is the best solution for XML documents.

5. Create an E-Commerce Web Site

The business may be small - but the Internet lets the think big. Whatever product or service the business offers, the Internet levels the playing field and lets the compete with bigger businesses, reaching customers around the world that can conveniently buy from the online storefront 24 hours a day. The first step toward e-commerce is selecting the name of the site. The Web address (also called a URL or "domain name"), tells customers are the web address and how to find this web on the Internet. It is the core of the Internet identity - the online brand. And because no two parties can have the same Web address, the online identity is totally unique. Let the name of the e-commerce site is: http://www.sahabcomm.com. Step two is building a User-Friendly Site that includes: Plan the site carefully, choose the right site building tools, carefully research the own favorite e-commerce sites, make it easy for customers to navigate the site, keep things simple, and keep download times short.

Step Three: Set up the Web Server - or Select an ISP to Host the Site the Web site is a series of files that reside on a special computer, called a Web server, connected to the Internet. For customers to visit the site, they must actually connect to that Web server via the Internet and view the files. Web servers and the Internet connections that link them to visitors must be fast and powerful enough to quickly respond to all the visitors' requests to view the site.

6. The designed protocol

Electronic commerce operations rely on cryptographic protocols, mutually subscribed series of steps involving the exchange of messages between principals [7]. Weaknesses in encryption contribute to vulnerabilities. But more serious are subtle design flaws [8]. Therefore, it is important to develop techniques for comprehensive protocol analysis, especially to verify security properties [9][10][11]. This includes complex messages and the knowledge and behavior of principals. In this protocol, the merchant have its multiple banks and the customer has it's bank. Also there is a bank that modeled to act as a trusted bank between the merchant and the bank. The main function of the trusted bank is checking the sending messages between the merchant and the customer that passed through it and making forwarding to the destination one if the contents for the specific message are valid and make correction for the message that has invalid information in its content.

A. The propose protocol

The Modeling of an electronic commerce protocol for a trusted banking is described as following (see the figure 1 bellow):

1. $C \rightarrow M$: $P_O, B_C, A_C, \{B_M, A_M, \$, Req\}_{Kc}$
2. $M \rightarrow B_M$: $B_C, A_C, \{B_M, A_M, \$, Req\}_{Kc}$
3. If $B_M = B_C$ then goto 7

   Else

$B_M \rightarrow B_T$: A_C,$\{B_M, A_M, \$, Req\}_{Kc}$

4. $B_T \rightarrow B_C$: A_C,$\{B_M, A_M, \$, Req, M\text{-}check\}_{Kc}$
5. $B_C \rightarrow B_T$:$\{A_C, A_M, \$, Req\}_{KBC,BM}$
6. $B_T \rightarrow B_M$: $\{A_C, A_M, \$, Req, C\text{-}check\}_{KBC,BM}$
7. $B_M \rightarrow M$:$\{B_C, A_C, \$, Req\}_{KM}$
8. $M \rightarrow C$:$\{P_O, B_M, A_M, \$, Req\}_{KM}$
9. $C \rightarrow M$: $P_O, B_C, A_C, \{B_M, A_M, \$, Req, PAY\}_{Kc}$
10. $M \rightarrow B_M$: $B_C, A_C, \{B_M, A_M, \$, Req, PAY\}_{Kc}$
11. If $B_M \neq B_C$ then

    $B_M \rightarrow B_T$: A_C, $\{B_M, A_M, \$, Req, PAY\}_{Kc}$

    Else goto 12
12. $B_T \rightarrow B_C$: A_C,$\{B_M, A_M, \$, Req, PAY, M\text{-}check\}_{Kc}$
13. $B_C \rightarrow B_T$:$\{A_C, A_M, \$, Req\}_{KBC,BM}$
14. $B_T \rightarrow B_M$: $\{A_C, A_M, \$, Req, C\text{-}check\}_{KBC,BM}$
15. $B_M \rightarrow M$:$\{B_C, A_C, \$, Req, PAID\}_{KM}$
16. $M \rightarrow$:$\{Req\}_{KR}$

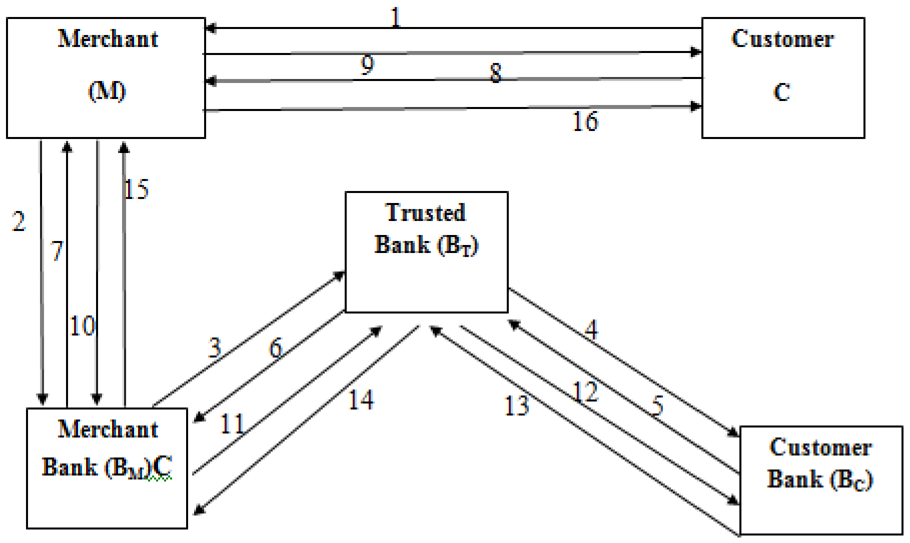

Figure (1): The Sequence of the designed protocol.

<u>B. In this paper, the used notations as following:</u>

Notation	Meaning
C	Customer.
M	Merchant.
P_O	The product order.
B_C	Bank of customer C.
A_C	Account number of customer C.
K_C	The confidential key corresponding to an account of customer C.
B_M	Bank of merchant M.
A_M	Account number of merchant M.
$	The type and amount of currency.
Req	The request of customer C
B_T	The trusted bank.
M-check	The B_T told B_C that the last message from B_M had been checked and it is valid message.
C-check	The B_T told B_M that the last message from B_C had been checked and it is valid message.
$K_{Bc,BM}$	The common key between bank of customer C and merchant M.
$\{x\}_{Ki}$	Message consisting of x encrypted in the key Ki.
$i \to x{:}j$	i sends a message consisting of x over the net to j.

7. The Web Service Design

The designed Web service has been implemented through software components that communicate using pervasive, standards-based Web technologies including HTTP and XML-based messaging. In this work, the Web services are that located on the merchant site are designed to be accessed by other client(s) and vary in complexity such as checking a banking account balance, and a customer relationship management. They are based on open standards and protocols as HTTP and XML-based protocols including SOAP, WSDL, and UDDI. First, the definition of the Web service in the form of a WSDL document that describes the service's location on the Web and the functionality the service is provided. Information about the service then will be entered in a UDDI registry, which allows Web service consumers to search for and locate the service they need. This step is optional but it is beneficial when a merchant wants its Web services to be discovered by internal and/or external service consumers. Based on information in the UDDI registry, the Web service client uses instructions in the WSDL to construct SOAP messages for exchanging data with the service over HTTP. The main steps required the design of the web services are summarized in the flow diagram shows in figure (2).

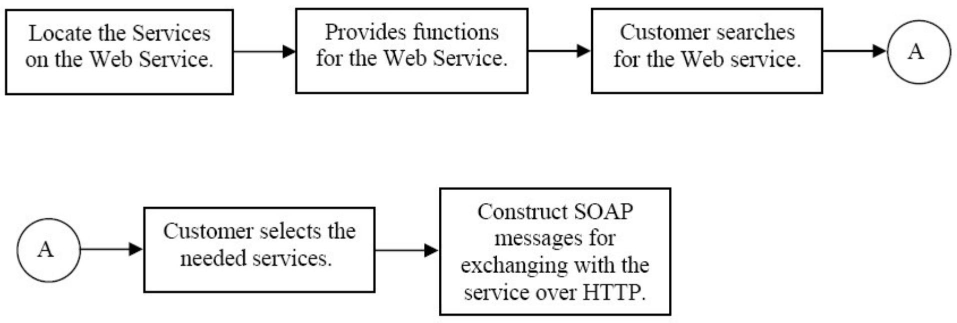
Locate the Services on the Web Service.
Provides functions for the Web Service.
Customer searches for the Web service.
A
A
Customer selects the needed services.
Construct SOAP messages for exchanging with the service over HTTP.

Figure (2): The flow diagram for the design of the web service.

The used WSDL file is an XML document that describes a Web service using six main elements:

- **Port type:** groups and describes the operations performed by the service through the defined interface.
- **Port:** specifies an address for a binding, i.e., defines a communication port.
- **Message:** describes the names and format of the messages supported by the service.
- **Types:** defines the data types (as defined in an XML Schema) used by the service for sending messages between the client and server.
- **Binding:** defines the communication protocols supported by the operations provided by the service.
- **Service:** specifies the address (URL) for accessing the service.

8.1 SOAP Messaging

We'll be implementing the interface defined in the last section as XML messages. There have been many initiatives to provide a standard mechanism for XML messaging. One of these is Simple Object Access protocol (SOAP). SOAP is a method of accessing remote objects by sending XML messages, which provides platform and language independence. It works over various low-level communications protocols, but the most common is HTTP. The Resources section points to some other material on SOAP, but for purpose of this hands-on tutorial, I'll present SOAP by brief example.

- **The HTTP request header**

The first part of the SOAP HTTP request header is familiar territory.

POST /calendar-request HTTP/1.1
Host: uche.ogbuji.net
Content-Type: text/xml; charset="utf-8"
Content-Length: 507

- **The SOAPAction header**

The only addition to the HTTP header for SOAP is the SOAPAction.
SOAPAction: http://uche.ogbuji.net/soap-example

This header is intended for firewalls and other network infrastructure that are aware of SOAP, especially for filtering and routing purposes. The value is a URI which identifies the action requested by this message. Note that there hasn't been a great deal of work on specifying such issues as security, firewall processing and routing of SOAP requests. Unfortunately, SOAP's popularity has to some extent outrun the development of its infrastructure, but these issues are currently being discussed in layers above SOAP.

- **The SOAP envelope**

The HTTP request body contains the SOAP request itself. This is wrapped in a SOAP envelope, which contains metadata important for understanding the request. The structure is as follows:
<SOAP-ENV:Envelope

```
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<!-- SOAP body goes here -->
</SOAP-ENV:Envelope>
```

The SOAP envelope is in the XML namespace prescribed by the specification: *http://schemas.xmlsoap.org/soap/envelope/.*

The envelope must specify how to interpret the SOAP body. This is accomplished with the *SOAP-ENV:encodingStyle* attribute. The value is a URI indicating a specification for the structure of the body.

- **SOAP encodings**

The envelope specified the encoding for the body as *http://schemas.xmlsoap.org/soap/encoding/.*

This is a method for structuring the request that is suggested within the SOAP spec itself, known as the SOAP serialization. It's worth noting that one of the biggest technical complaints against SOAP is that it mixes a specification for message transport with a specification for message structure. You needn't feel constrained to use the SOAP serialization encoding style. Other possible encoding styles include Web Distributed Data Exchange (WDDX), XML Remote Procedure Call (XML-RPC), Resource Description Framework (RDF), or just a custom XML structure. The latter is probably good enough if you're not worried about whether an attribute value of "1" is supposed to be interpreted as a string or an integer, for example. See Resources for information on these encoding styles.

- **The SOAP body**

There is one more layer of SOAP element enclosing the actual elements of the specialized event calendar requests.

```
<SOAP-ENV:Body>
<!-- User request code here -->
</SOAP-ENV:Body>
```

- **The SOAP Elements**

The used SOAP messages are XML documents that contain the following elements:

- **Envelope:** specifies that the XML document is a SOAP message; encloses the message itself.
- **Header (optional):** contains information relevant to the message, e.g., the date the message was sent, authentication data, etc.
- **Body:** includes the message payload.
- **Fault (optional):** carries information about a client or server error within a SOAP message.

Data is sent between the client(s) and the Web service using request and response SOAP messages, the format for which is specified in the WSDL definition. Because the client and server adhere to the WSDL contract when creating SOAP messages, the messages are guaranteed to be compatible. Figure (3) and figure (4) illustrate what is the complete cycle of a SOAP message request and response in addition SOAP building blocks that they pass.

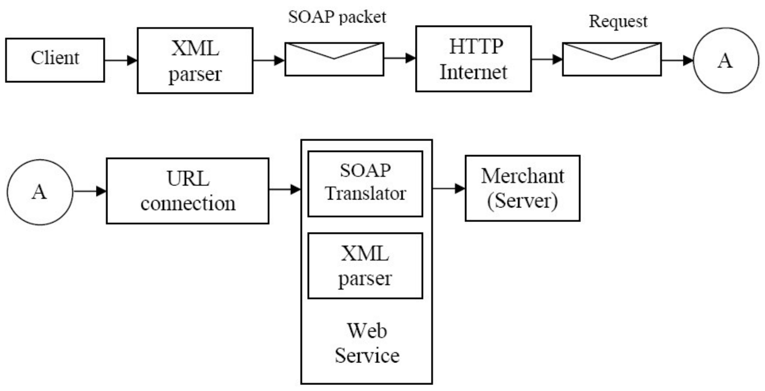

Figure (3): The complete cycle of a SOAP message request.

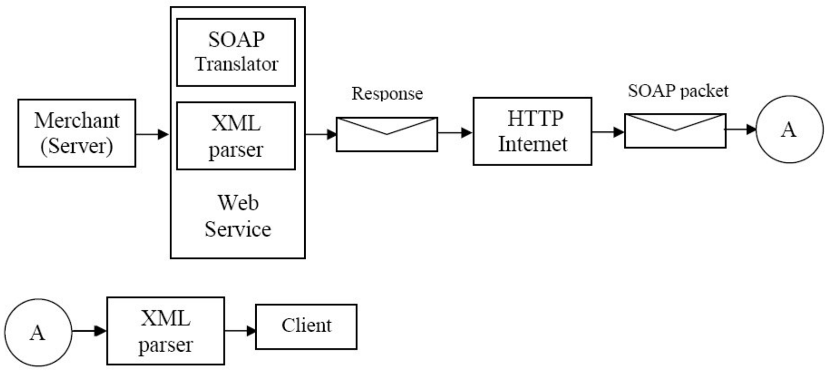

Figure (4): The complete cycle of a SOAP message response.

8. Modeling the Customer Process:

The customer browses the products or services catalogue hosted on the merchant site and fills in the order on demand, and could modify the order information. If the customer is not satisfied with the product or service that has chosen, the transaction will be cancelled. Once the purchase order has been sent to the merchant, the payment information encapsulated in the order would also be sent to the merchant. Once the payment information is sent to the bank, this will invoke three cases:

(1) The transaction will abort when the customer's funds are insufficient.

(2) If the customer's funds are available, it will be held by customer's account after the first payment operation.

(3) If technical errors occur after sending the payment information by the customer, the transaction will abort. Once the merchant has received the funds information from its bank, the procedure proceeds to the next step and notify the customer to confirm the product or service which he has ordered. Lastly the customer should send the payment message, and the transaction would be over.

9. Modeling the Merchant Process:

On the merchant side, the merchant is waiting for receiving payment information from the customer. There are three possible cases related to this process:

(1) The transaction would abort if the customer has insufficient fund.

(2) The procedure proceeds to notify the customer to confirm the product.

(3) After the merchant receives the token from the customer to confirm the order, the merchant in turn forwards the token to the bank, and then a procedure is initiated to transfer the funds from the customer's account to the merchant's bank account.

10. Modeling the Bank Process:

The customer's bank sends and receives all messages to/from the trusted bank. The merchant's bank receives the payment information from the merchant, and then make forwarded to the trusted bank that will be checks the payment information of the customer. There are three possible cases:

(1) The transaction would abort if the customer's account has insufficient fund.

(2) If the funds are available, the requested amount will be held on and the funds information will be feed back to the merchant bank. When the merchant bank account receives the payment message from the customer, the funds in the customer account will be transferred to the merchant bank account, and the payment operation terminates.

(3) If the merchant account receives the message abort from the bank of the merchant, the transaction is cancelled.

11. Result:

The designed model is based on a centralized storage and access model that called trusted bank. There is a large need for a central server (trusted bank) for user registration, aggregation and processing of data. Any use of a centralized server would be against the intruders. Since the collection, linking and processing of information is completely distributed, the proposed model is more failsafe and robust. As is known for its e-commerce protocol designed in this research, the input is to fill the registration form by the purchaser and sent to the bank and assigned after checking and verifying the information entered on her head, and method of payment and confirmation of the payment card validity as well as in terms of whether used by the buyer The other or not. Those inputs are invisible to all but the bank's authorized seller is waiting for either a signal from the bank then authorized the seller to send goods requested by the buyer or his request after he was required to pay the amount of the bank authorized. Those goods and demand is the outcome of a result for the buyer. The designed protocol as a whole contains many of the outputs of the coders of the technical aspects of innovation and is projecting it on the home page of the seller or merchant. The programming languages that add security and aesthetic aspects, flexibility and many others in addition to the flexibility and security that was built by the Protocol, which was built in this paper.

12. Conclusions

In this work a proposed e-commerce protocol that has some desirable features. First, it provides fair exchange under all circumstances. Second, the protocol does not require any manual dispute resolution in case any party behaves unfairly. Third, the protocol does use a trusted bank that lies between the merchant's bank and customer's bank. Fourth, the protocol allows the customer to be confident that he is paying for the correct product before actually paying for it. Fifth, the protocol was designed using the web service model that add reliable and flexible features to overall transactions. One future work is evaluating the correctness of the protocol using formal methods of software verification like model checking [5]. The second Future work that will increase the efficiency and add security features to the system protocol.

13. References

[1] Wenli Wang, Zoltan Hidvegi, Andrew D. Bailey Jr. and Andrew B. Whinston. Bprocess design and assurance using model checking, IEEE Computer, pp.48-53, October 2000.

[2] G. Lowe and A. W. Roscoe. Using CSP to detect errors in the TMN protocol. IEEE Transactions on Sofiate Engineering, 23:659-669,1997.

[3] Zhigang Wu, Binxing Fang, Peng Sun and Yaping Li, A Secure, Atomic electronic commerce protocol and its formal verification. Journal of computer Reseazh & Development, VO.3~7 ,N0.7, July 2000, pp. 869-873.

[4] Vincent Gregoire, Authentication Solutions for e-Banking Services, Gemalto, the Gemalto logo, are trademarks and service marks of Gemalto and are registered in certain countries, October 2008.

[5] Kun-Lung Wu and Philip S. Yu, " Report on Second International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems ", IBM T.J. Watson Research Center 30 Saw Mill River Road Hawthorne, NY 10532

[6] John W. Shipman, New Mexico Tech Computer, Constructing a DTD for XML Center, January 2008.

[7] Heintze, N., Tygar, D., Wing, J. and Wong, H. (1996) Model checking electronic commerce protocols. Proceedings of the I996 USENIX Workshop on Electronic Commerce.

[8] Lowe, G. (1996) Some new attacks upon security protocols. Proceedings of the Ninth IEEE Computer Security Foundations Workshop.

[9] Burrows, M., Abadi, M. and Needham, R. (1990) A logic of authentication. ACM Transactions on Computer Systems, 8(1), 18-36.

[10] Kindred, D. and Wing, J. (1996) Fast automatic checking of security protocols. Proceedings of the I996 USENIX Workshop on Electronic Commerce.

[11] Lowe, G. (1996) Some new attacks upon security protocols. Proceedings of the Ninth IEEE Computer Security Foundations Workshop.