A Kerberos based Framework for Electronic Tender Processing

Rashi Kohli
Computer Science, Amity University

Divya Sharma
Computer Science, Amity University

Santar Pal Singh
Information Technology, KIET

ABSTRACT

Proposed system structure can be used in any World-Wide Web technology to make possible implementation of a secure electronic tender processing system. This System offers possibilities considering both cost-effectiveness and security. Providing mutual authentication [11,12] is the correct way to make possible secure communication and transmission of data among parties with reliability, confidentiality & security in WAN, MAN, LAN etc technologies used in Electronic Tender Processing & this can be achieved by using Kerberos as an Authentication services. In this paper the system structure and communication protocol of a system utilizing these possibilities are proposed, that should be profitable when designing a final implementation.

Keywords: Confidentiality ,Integrity, Non-repudiation ,Security ,cryptography, Authentication , Hypertext Transfer Protocol(HTTP), Secure Socket layer(SSL), WAN, MAN, LAN, World-Wide-Web(WWW).

1. INTRODUCTION

The goal is to capture the best possible offer to supply the services or goods in question,When something needs to encounter while making a request for tender. To procure this several steps have to be performed. Firstly, the tender invitation, should be written that captures the expected characteristics of the requested services or goods in a precise and unambiguous manner, Secondly, the tender invitation has to be distributed in such a way that parties which are potentially interested in responding to the same will become aware of its existence.Thirdly, the parties interested in delivering the requested services or goods, henceforth referred as Suppliers. They have to write tenders capturing their offers. Fourthly, the tenders have to be delivered to the party requesting them which is the buyer, preferably in a secure way. Here, security implies confidentiality, integrity, and non-repudiation. Fifthly, the buyer has to decide whether any of the submitted tenders is fulfilling the requirements and, if this is the case, buyer has to decide which suppliers is efficient enough to get the contract The tender process is followed agreed upon and signed by the buyer itself and the supplier so as to close the deal.

In this paper it aims at designing a secure electronic tender system .In this paper World-Wide-Web (WWW) technology is utilized to support the execution of steps two and four of the tender process described above.

1.1 PROBLEM FORMULATION

Designing a secure electronic tender system demands careful consideration of the requirements implied by the terms which are secure and electronic. In this case, electronic means that WWW technology should be used in order to avail benefit from the connectivity and low distribution costs of the Internet. Thus, a Framework for distribution of tender invitations and tenders via the Internet has to be created. Being secure, on the other hand, requires that the confidentiality, integrity, and non-repudiation characteristics of the documents should be properly protected. Confidentiality [1] is usually not an issue for Tender invitations. However, in some cases the mere fact that the buyer would like to purchase a Specific service or goods which refers to the sensitive information. For tenders, on the other hand, confidentiality is an essential component from the perspective of both the buyer as well as the supplier. Integrity is undeniably a central issue considering all the involved documents and parties, since the contents of a received document should be trustworthy. Furthermore, non-repudiation is required in order for a party to take advantage of the offer offered by another party. Thus, a protocol with security mechanisms guaranteeing these characteristics of the documents has to be designed.

1.2 CONTRIBUTION

The main contributions of this work are:

(a) The design of a system structure and a protocol fulfilling the requirements specified in the Section 1.1 above.

(b) Explaining the use of the proposed protocol.

(c) A discussion about the strengths and weaknesses of the prototype implementation.

2. BROAD SURVEY OF THE WWW AND SECURITY ENHANCEMENTS WE USE

The survey consist of four sections

Kerberos: Kerberos is used to grasp authentication of tenders and tender invitations.

WWW-servers: basic functionality. The server shall serve two types of requests from the clients.The clients want to read data (a static data file or some dynamically generated data) or the clients want to submit data (to be stored in a file or to be input to an application). The basic means to implement a security policy, i.e. control in which client (or preferably which user) is authorized to read or submit, are quite rude.

WWW-clients: basic functionality of clients, mainly browsers.

WWW-security: The de facto enhancements for better security in WWW-servers and clients.

2.1 THE KERBEROS AUTHENTICATION SERVICE

Kerberos is a distributed authentication service that allows a process (a client) running on behalf of a principal (a user) to prove its identity to a verifier (an application server, or just server) without sending data

across the network this might allow an attacker or the verifier to subsequently impersonate the principal. Kerberos optionally provides integrity and confidentiality for data sent between the client and server. Kerberos was developed in the mid-'80s as part of MIT's Project Athena [2]. As use of Kerberos spread to other environments, changes were needed to support new policies and patterns of use. To address these needs, design of Version 5 of Kerberos (V5) begins in 1989. Though V4 still runs at many sites, V5 is considered to be standard Kerberos [7].

2.2 WWW-SERVERS

The communication protocol mainly used on the World-Wide Web is HTTP, Hyper Text Transfer Protocol. In its simplest form it is used by a client to request data files, of different multimedia types, from a WWW-server. These requests should be served without the client logging in or otherwise client has to authenticate itself for the same. The data request either consists of a file name, which results in transmission of this particular data file, or it consists of a program name, which results in transmission of the output of the program. In the latter case the client could also send small amounts of data as input to the requested program.The HTTP protocol is layered right above the TCP protocol. In the basic version, HTTP version 1.0, there is no persisting TCP-sessions.

Each request requires its own session. So the basic HTTP 1.0 is quite simple. It mainly consists of three commands from the client - HEAD to get information, e.g. multimedia type, concerning the file, GET to also get the data file and POST to input data to a process in the server, e.g. to append data in a database.

There are extensions in later versions, such as HTTP 1.1, but they are not implemented everywhere. Examples are PUT to send a data file from the client and methods to achieve some rudimentary authentication by sending a password unencrypted. The security functions are almost non-existent in a WWW-server that only implements basic HTTP 1.0. Often the only restriction is that you can only request a file or a program that is stored in a specific branch of the file system in the server. Common extensions to somewhat improved security are for instance Access Control Lists. The ACL could be hard coded in the configuration file of the server or it could be placed in a file, for instance named. Htaccess , for each node in the branch. The authentication of the client is all the same imperfect, since it is often based on the IP-address of the client or on a password sent as clear text.

Similar deficiencies exists when the client POSTs input data to a server program. The program doesn't get authenticated information about which client is issuing the request. The program could be a specific application program with the POSTed data as input. It also could be a script that should be interpreted by a general script interpreter, for instance Perl. Some servers can interpret scripts themselves without running a separate script interpreter. Obviously these programs and scripts must be written with great care otherwise security will be compromised which will act as disadvantage to the system.

2.3 WWW-CLIENTS

The WWW-client uses the HTTP-protocol to request data from the server or to submit data to a program or script controlled by the server. The client could be any type of application. The most common application is the browser which isused to display the requested data, that is embedded in the data which is the control information, written in the standardized HTML-language that tells the browser how data should be displayed. Due to the enormous popularity of the WWW, and the many types of multimedia information, some browsers have become huge programs trying to integrate a lot of peripheral functions, for instance electronic mail. But you can also find smaller browsers focusing on the core task, to display HTML(5)files.A simple browser can display a HTML-file. It can also display a form where the user can type data that the browser POSTs to the server. To extend this basic functionality there was a need for a scripting language that could interpret scripts embedded in the HTML-file. The de facto language is JavaScript [6] which is implemented in most browsers. One example of use of JavaScript is to describe some calculations to do on data from a form before they are POSTed in order to detect typos. One issue with JavaScript is that it is not properly standardized. The next level of extended browsers has implemented languages with more functionality than JavaScript. The most well known language is Java. Many browsers have hooks where you can add extensions called plug-ins. Such a plug-in could be an interpreter or it can be another scripting language, e.g. Tcl/Tk or Perl, or it could be a plug-in for some type of multimedia, e.g. video. As an alternative to plug-ins the browser could launch a separate helper application to which it submits the data. The security implications of these different extensions are equally serious as those on the server side and the need for authentication is evident. The functionality in scripts and plug-ins must furthermore be properly restricted. From JavaScript it should be for instance not be possible to access the file system or the network at all. From Java ,you can grant access in a restricted way, within a 'sandbox'. From extensions built on Microsoft's ActiveX you have no restrictions other than authentication. Code which comes from an authorized origin can do anything. For a discussion on different aspects of these matters, mobile code, see (Persson 1998).

2.4 WWW-SECURITY

In the previous sections we noticed that the WWW-security should be enhanced in many ways. This could be done at two levels. At end-to-end level where the applications, in the clients and the server respectively, control encryption, authentication and access policy themselves. To achieve this the necessary function is encryption. The enhancements could also be at the transport layer. There have been proposals for extensions to the HTTP protocol but the de facto extension is SSL, Secure Socket Layer, in an intermediate layer just above TCP layer which means that it can be used by other protocols besides HTTP. The standardized version of SSL is named TLS. The SSL-protocol consists of two parts, one handshake part and one transmission part. The handshake part is the opening of a persistent session within which the transmission can be split in many TCP-connections. The handshake has two results, one mandatory and one optional. The mandatory result is that the client and the server, using a negotiated method, build up a mutual master secret. The optional result is that the client or the server, or both of them, authenticates itself. To achieve this it sends a list of certificates, up to the root certificate, that the other party can verify. The authentication is at the SSL-level only. There is for instance no standardized way to hand over the certificates to the application.

In the transmission part of SSL [4] the two parties each computes a key based on the master secret, and negotiates an encryption method, to be used for transmission encryption. This can be done several times within a session. This incorporates towards the confidentiality at the transmission level, but data is decrypted before it is handed on to the application. In SSL-capable WWW-browsers the convention is that if the address of the link starts with https:// instead of http:// then SSL should be used. The most common use of SSL, for instance in systems for shopping on the Internet, is to only use transmission encryption. In this case you get confidentiality on the Internet but you don't necessarily get authentication, although most WWW-servers send a list of certificates that makes server authentication possible.

3. PROPOSED SYSTEM STRUCTURE AND COMMUNICATION PROTOCOL

In order to construct the secure tender system a protocol will have to be specified. The protocol is the definition of how the communication between the parties will be structured in order for the system to work. The protocol set up must handle at least two problems.

(1) Authenticity [5] of the offers placed on the server by the buyer.

(2) Confidentiality and authenticity of the tenders placed on the server by the provider. This is accomplished through the use of encryption and Kerberos as detailed below.

3.1 PROPOSED SYSTEM STRUCTURE

The entire system will consist of a web server on which the invitations and tenders will be located, the world- wide - web and at least one other computer. The owner of the system, the buyer, will be putting the tender invitations directly on the server. The prospective supplier will connect his computer with the server through the Internet. The encryption [8] is handled through the use of the SSL protocol in the contacts between the computers. This is done automatically when the supplier is using one of the reasonably modern browsers and hence is a very practical solution as it is transparent to the user and requires no knowledge of encryption techniques on their part. Since this feature was used as-is we do not take credit for it and will not discuss it further in this paper which is made with Netscape built-in features and require no knowledge of programming on the part of the users. It is however not automatic and is included in the protocol description below

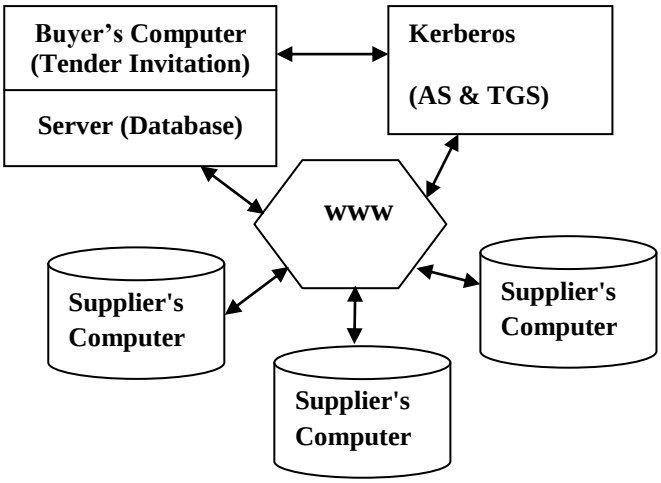

Fig.(a) System structure for Electronic Tender Processing

3.2 ENCRYPTION

The encryption part used in the protocol will be handled by using SSL for all the transactions. The guarantee of the correctness of the cryptographic [13,14] implementation will therefore rest on the providers of the browser and web server.

3.3 PROTOCOL DEFINITIONS:

B will denote the buyer's Identity.
S will denote the Supplier's Identity.

TGC will denote the ticket granting server. AS will denote the Authentication server. K_B will denote the Buyer's secret key.

K_{TGS} will denote TGS's secret key .
K_S will denote the Supplier's secret key.

$K_{BS(t)}$ will denote the shared key for Buyer and Supplier at time (t).

$K_{BS(t+1)}$ will denote the shared key for Buyer and Supplier at time (t+1).

$K_{ses\ will}$ denote the session key.

3.4 Communication Protocol Structure (encryption excluded).

Kerberos involves three servers in addition to.

(1) Buyer (A client workstation)

(2) AS (Authentication Server): Verifies user during login, shares a secret password with every user.

(3) TGS (Ticket Granting Server): Issues proof of identity tickets, its job is to issue tickets that can convince the real server that the bearer of a TGS ticket really is who he or she claims to be.

The Operation of Kerberos Authentication Protocol

Suppliers: Actually does the work Buyer wants performed To start a session, Buyer sits down at a arbitrary public workstation and types his name .The Buyer sends his name to the AS in plaintext ,as shown in Fig.(B) what comes back is a session key and a ticket ,K_{TGS} (B, K_{Ses}), intended for the TGS. These items are packaged together and encrypted using Buyer's Secret key K_B ,so that only Buyer can decrypt them. Only when message 2 arrives, does the workstation asked for Buyer's password.

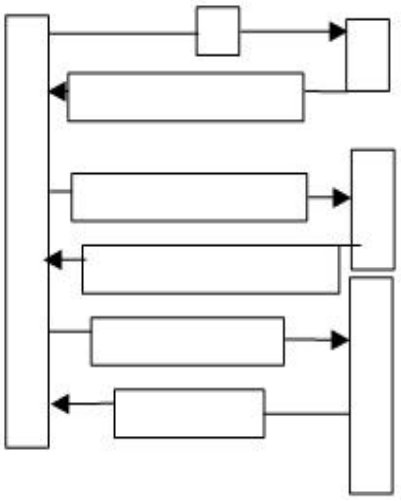

Figure- (b)

The password is than used to generate K_B in order to decrypt message 2 and obtain the session key and TGS ticket inside it. At this point, the workstation overwrites Buyer's password ,to make sure that it is only inside the workstation for a few milliseconds at most . if Hacker tries logging in as Buyer ,the password he typed will be wrong and the workstation will detect this because the standard part of message 2 will be incorrect.After Buyer willlogs in, Buyer may tell the workstation that Buyer wants to contact Supplier. The workstation then sends message3 to the TGS asking for a ticket to use with Supplier .The key element in this request is K_{TGS}(S ,KSes),which is encrypted[5,8] with the TGS's secret key and is used as proof that sender really is Buyer .The TGS responds bye creating a session key ,K_{BS} ,for Buyer to use with supplier. Two versions of it are send back. The first is encrypted with only Ks ,so Supplier can read it. Now Buyer can send K_{BS} to Supplier to establish a session with him .This exchange is also time stamped. The response is proof to Buyer that he is actually talking to Supplier not to Hacker. After this series of exchanges, Buyer can communicate with Supplier under cover of K_{BS}. If Buyer wants to talk to another server S1, he just repeat message3 to the TGS ,only now specifying S1 instead of S. The TGS will promptly respond with a ticket encrypted with K_{S1} that Buyer can send to S1 and that S1 will accept as proof that it came from Buyer.

4. CONCLUSION

In this paper a secure electronic tender system [9,10] has been proposed and evaluated. The proposed system can be used to decrease the effort that has to be put into the tender process and, thus, increase its cost- effectiveness. Moreover, the security issues [3,15] of confidentiality, integrity, and non-repudiation are handled. Except for an extension handling the contract phase following the tender process, the possible future extensions and improvements relate to the security and user convenience of the proposed system structure. A more tightly integrated implementation is required for non-experienced (or not security-aware) users. For example, the identity of the suppliers could be verified automatically by the system upon arrival to the WWW-server. On the other hand, by separating the tools for tender invitation posting from the WWW-server the security will be improved, since the corresponding CGI-script will not be accessible from the outside any longer.

5. REFERENCES

[1] Heimdal and Windows 2000 Kerberos-how to get them to-Westerlund,Danielsson (2001)

[2] Asad Amir Prizada and Chris McDonald, "Kerberos Assist Authentication in Mobile Ad-hoc Networks",School of Computer Science & Software Engineering,The Univercity of Western Australia,Sep 2004

[3] Applied Security Technologies, Smart Card Lifecycle Security Guidelines, Issue 1.0, dated 19 May 2003.

[4] OpenSSL. http://www.openssl.org/.

[5] Schapper, P. R. "Authentication: The Art of Discovering Whether Appearances are True or False" Proceedings of the Electronic Government Procurement Conference, Manila, Asian Development Bank, Inter-American Development Bank, World Bank., Oct 2004

[6] Netscape, Signing Text from JavaScript, 2004. http://developer.netscape.com/docs/manuals/security/sgntxt/index.htm

[7] Douglas E.Engert, "Self Imposed Limitations of Kerberos",Argonne National Laboratory,April 2004.

[8] (8) Joan Daemen and Vincent Rijmen. Rijndael, the Advanced Encryption Standard. Dr. Dobb's Journal, 26(3):137{139, March 2001

[9] Information Security Committee Electronic Commerce Division, Section of Science & Technology Law, American Bar Association, "PKI Assessment Guidelines", Guidelines to help assess and facilitate interoperable trustworthy public key infrastructures, PAG v0.30 Public Draft for Comment, June 2001.

[10] MasterCard. Secure Payment Application (SPA). http://www. mastercardintl.com/.

[11] Burr-Donna, W. E., Dodson, E., Polk, W. T. "Electronic Authentication Guideline" .Recommendations of the National Institute of Standards and Technology. September 2004.

[12] Clarke, R. Authentication: A Sufficiently Rich Model to Enable E-Business.

[13] http://www.anu.edu.au/people/Roger.Clarke/EC/AuthModel.html

[14] Clarke, R. The Reinvention of Public Key Infrastructurehttp://www.anu.edu.au/people/Roger.Clarke/EC/PKIReinv.html Committee on Government Reform, House of Representatives. "Advances and Remaining Challenges to Adoption of Public Key Infrastructure Technology", Report to the Chairman, Subcommittee on Government Efficiency, Financial Management and Intergovernmental Relations, February 2001.

[15] Directorate for Science, Technology and Industry Committee for Information, Computer and Communications Policy, "Summary of responses to the survey of legal and policy frameworks for electronic authentication services and e-signatures in OECD member countries", July 2004.

[16] www.oecd.org/sti/security-privacy

[17] e-Government Interoperability Framework,Part2, Version 5.1, October 2003