Journal of the Nigerian Association of Mathematical Physics
Volume **18 (May, 2011), pp 423 – 428**

A PROTOTYPE SECURITY MECHANISM FOR XML DOCUMENTS AND SOAP MESSAGES

[1]Evwiekpaefe A. E., [2]Irhebhude M. E., and [3]Adedayo, D. A.
[1,2]Department of Mathematics/Computer Science,
Nigerian Defence Academy, Kaduna, Nigeria
[3]Department of Mathematics, Statistics and Computer Science,
Kaduna Polytechnic, Kaduna, Nigeria

Abstract

Cyber crime is a recurrent phenomenon on the internet despite various preventive measures towards these activities. As web servers develop means to protect their data and/or services, hackers are improving everyday on their skills to break these measures. It is in light of this that we developed a model that demonstrates how to best safeguard log information so that when such cyber crime takes place, there can be a way to track such criminals even when they succeed. It also shows how we can better protect our log information from unauthorized users of the system so that as accesses are made, protection and avoidance mechanisms can be enforced.

Keywords: Model, Security, SOAP, Web services, XML

1. Introduction

Simple Object Access Protocol (SOAP) was developed by Microsoft, DevelopMentor, and Userland Software and proposed as an XML protocol to the World Wide Web Consortium (W3C). The name reflected the idea that SOAP would be used to express serialized object graphs, enabling object-oriented systems to perform functions such as remote procedure calls while preserving objects and their relations. However, in the W3C's latest working draft (version 1.2), SOAP became the name and is no longer an acronym. This reflects a shift in thinking about SOAP from a serialization framework for object-oriented systems to a more general XML-based messaging paradigm, where the messages do not necessarily contain objects [3].

SOAP is a simple, flexible and extendible mechanism for exchanging structured data, primarily designed for providing an RPC (Remote Procedure Call) mechanism on top of the widely used internet standards of XML (Extensible Markup Language) and HTTP (Hypertext Transfer Protocol) [5].

Furthermore, according to [4], SOAP is both language and platform independent and can easily get around firewalls. It defines a set of rules for structuring messages that can be used for simple one-way messaging but is particularly useful for performing RPC-style (Remote Procedure Call) request-response dialogues. It is not tied to any particular transport protocol, though HTTP is popular, nor is it tied to any particular operating system or programming language so theoretically the clients and servers in these dialogues can be running on any platform and written in any language as long as they can formulate and understand SOAP messages. As such it is an important building block for developing distributed applications that exploit functionality published as services over an intranet or the internet

Web services are application components whose functionality and interfaces are exposed to potential users through the application of existing and emerging web technology standards including XML, SOAP, WSDL and HTTP. In contrast to websites, browser-based interactions or platform-dependent technologies, web services are inter-computer services via defined formats and protocols, in a platform independent and language neutral manner.

SOAP is rapidly becoming the standard for building web services and connecting disparate systems in a loosely coupled fashion with complete platform independence. However, some of the features that make SOAP attractive, such as its flexibility and its compatibility with HTTP, also provide opportunities for security breaches [3]. Again, because no security is required in HTTP, XML, or SOAP, it is a pretty simple bet that different people will burgle any embedded security in different ways, leading to different holes on different implementations. SOAP is going to open up a whole new avenue for security vulnerabilities [1].

Moreover, ensuring the security of web services through a comprehensive security model is critical to both organizations and their customers. Unrestricted access and lack of an audit trail results in increased threat to the integrity and confidentiality of a business. The current web services standards that have achieved industry consensus (SOAP, UDDI and WSDL – all built on a foundation of XML) do not offer any specific provisions for security [6].

Hence, in this research our security model demonstrates how we can better protect our log information from unauthorized users of the system.

Corresponding authors: ***Evwiekpaefe A. E ., E-mail:*** contact_abraham@yahoo.com, Tel:+2348035600524

Objective of study

The objective of this study is to develop a prototype model (payroll system software model) that will demonstrate how to better enhance the security requirements of SOAP messages for more efficient service delivery.

Significance of study

There are several security measures in place to ensure safe operations on the internet. Despite these measures, we still have internet crimes, like the "yahoo yahoo guys", hacking of web servers, infecting of systems with viruses and worms, the list is endless. Even when these crimes are committed, offenders go free without being detected. Therefore, this work presents a better way to ensure the security of log information on the internet.

2. THEORETICAL MODEL

Analysis of the security requirements adopted by existing system requirements (IBM commercial server)

Authentication

Authentication is the process of verifying that users or applications are who they claim to be. The user authentication process is always performed under secured socket layer (SSL). This ensures that a third party using network-sniffing programs cannot snoop on the network when a user submits a password [2].

Authorization

Authorization is the process of determining whether a user can perform a specific operation on a resource. Authorization is determined from the access control policies governing the server [2].

Access Control Policies

An access control policy is a rule that describes which group of users is authorized to perform particular activities on your site. These activities can range from registration, to managing auctions, to updating the product catalog, and granting approvals on orders, as well as any of the hundreds of other activities that are required to operate and maintain an e-commerce site [2].

Audit trail

In computing, an audit trail is used to refer to electronic or paper logs that are used to track computer activity. For example, an employee might have access to a portion of a corporate network such as account receivable, but may not be authorized to access other portions of the system, such as payroll. If that employee attempts to access an unauthorized section by typing in passwords, this improper activity is recorded in the audit trail [2].

Confidentiality

Confidentiality is the process of protecting sensitive information from being deciphered by unintended recipients. Confidentiality is required when sensitive information flows from the user's browser to the server and back from the server to the user's browser [2].

Dataflow diagram of Soap request/response

Imagine we have a very simple corporate database that holds a table specifying employee id number, name and telephone number. You want to offer a service that enables other systems in your company or in a unit to do a lookup on this data. The service should return a name and telephone number (a two element array of strings) for a given employee id number (an integer). Here is a Java-style prototype for the service:

String[] getEmployeeDetails (int employeeNumber);

The SOAP developer's approach to such a problem is to encapsulate the database request logic for the service in a method (or function) in C or VB or Java etc, then set up a process that listens for requests to the service; such requests being in SOAP format and containing the service name and any required parameters. As mentioned, the transport layer might be HTTP though it could just as easily be SMTP or something else. Now, the listener process, which for simplicity is typically written in the same language as the service method, decodes the incoming SOAP request and transforms it into an invocation of the method. It then takes the result of the method call, encodes it into a SOAP message (response) and sends it back to the requester. Conceptually, this arrangement looks like this: see Fig. 1.

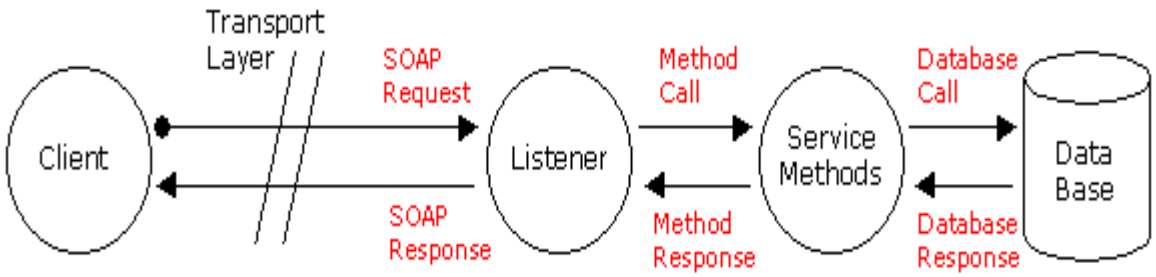

Fig 1: Conceptual view of SOAP operations *(Source:[4])*

While there are many different specific architectures possible for implementing this arrangement, for the purposes of illustration we will summarise one specific possibility.
The developer writes the service method in Java and connects to the database using an Oracle implementation of JDBC. The listener process is a Java Servlet running within a Servlet Engine such as Tomcat. The servlet has access to some Java classes capable of decoding and encoding SOAP messages (such as Apache SOAP for Java) and is listening for those messages as an HTTP POST. The transport is HTTP over TCP/IP. The client is an excel spreadsheet. It uses a VB Macro which in turn exploits the Microsoft SOAP Toolkit to encode a SOAP request and decode the response received. Here is a schematic of what that specific implementation looks like: see Fig 2.

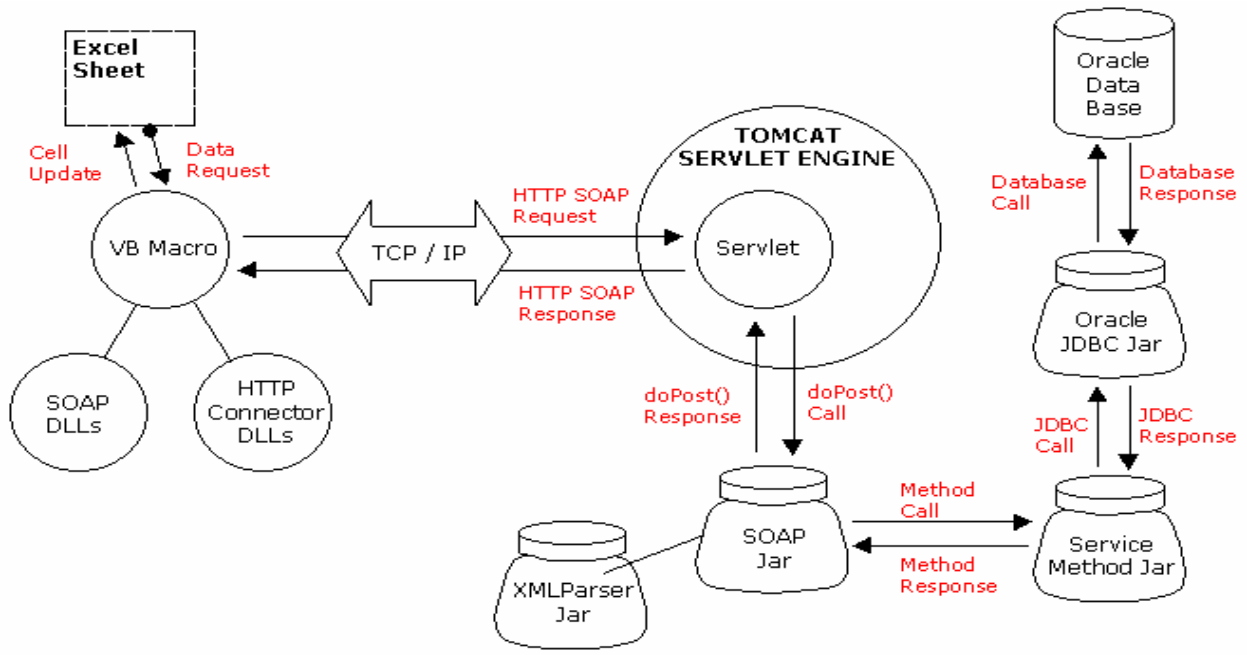

Fig 2: Schematic view of SOAP operations *(Source: [29])*

The client side of the VB Macro relies on both the Microsoft SOAP Toolkit (the SOAP DLLs) and a HTTP Connector interface. Such HTTP Connector DLLs are typically already installed as a part of Internet Explorer. On the server side you will notice that the SOAP package relies on some XML Parser to parse the SOAP messages.

Weakness of existing SOAP security requirements

The current security requirements suffer from the following:

a. For audit purpose, cyber criminals may decide to also clear their activities or modify it thereby compromising the audit record.
b. Because SOAP is stateless, cookie is used to identify every user. A criminal can also hijack a session from a particular user and commit a crime, thereby leaving the record in a compromised state.
c. HTTP is a stateless protocol; it does not maintain user information in a persistent way, so it asks the user to pass his user id and password for every new session with the server. [7]
d. The password sent to the server over the network is not encrypted, so it can be hacked by anyone and misused [7].
e. For server-side implementation, it always looks in the text file for username and password verification. It is very slow when it needs to check a very large set of data. It ultimately affects the system's performance [7].
f. This model uses text files to store the authentication information. This is one of the limitations of the basic authentication model that applies here also. The server takes a lot of time to verify the user name and password. An alternative mechanism is to use a database to store and retrieve usernames and passwords [7].

Requirements Specifications of the Prototype Model

The design is to address security concern of session hijacking and audit trail records, so that there can be effective auditing in case of any cyber crime of any kind. Even if a crime is committed, it can be detected and such perpetrators detected and prosecuted.

We adopted a prototype approach for the design and implementation of our proposed security model. The system consists of a client or user interface and a database which was integrated in such a way that the client can access the database from the interface. The client must have access to a PC with Graphical User Interface (GUI) capability to enable the usage of the software. Microsoft Access 2007 software database was used to hold all employees records for the demonstration. The model has administrator as well as users' access. The administrator has unlimited access to all menus and commands, whereas other users have limited functionality. The administrator coordinates expansion of the database when it becomes necessary or when the need arises.

The proposed system model was implemented to help complement the existing security requirements of the web server. The system model, if adopted by the W3C would help in the enforcement of an additional security requirement. The additional field involves an authenticated log (or cookies) access to prevent undue tampering of the system log or cookie to enable for effective auditing of the system tray. If a client fails to supply the right username and password after three attempts, the service shuts down. The system records all clients' activities on the server and if there be any form of hijacking, it is also recorded and given a password to prevent any hijacker from tampering with the log information.

Visual Basic 6.0 for Windows was used to develop the system which was used to demonstrate the security requirements. MS Access was used to design the database that was integrated or linked with the VB6 forms. VB6 was used to design all forms and codings of the forms for effective and efficient performance. Program codes are available on request.

3. RESULTS AND DISCUSSIONS

The model was designed and implemented in Visual Basic Programming Language. This model which is a Payroll System (The model is shown in figures 3 and 4 below) was designed with some security functionalities to explain what security requirement are to be embedded into the soap security requirement. The prototype model involves the enforcement of an additional security requirement. The additional field involves an authenticated log (or cookies) access to foil unwarranted tampering of the system log to facilitate effective auditing of the system tray. Once launched an authentication is needed to gain access into the system. This authentication has three attempts for a particular user to supply the right username and password, after which the system shuts down itself. Audit trail is the main features of this system. This system thus follow the audit trail left by a perpetrator since the system records all clients' activities on the server and if there be any form of fraudulent attack, it is recorded in the activity log and a password is given, to prevent any hijacker from tampering with the log information, thereby making it worthy for the audit tray.

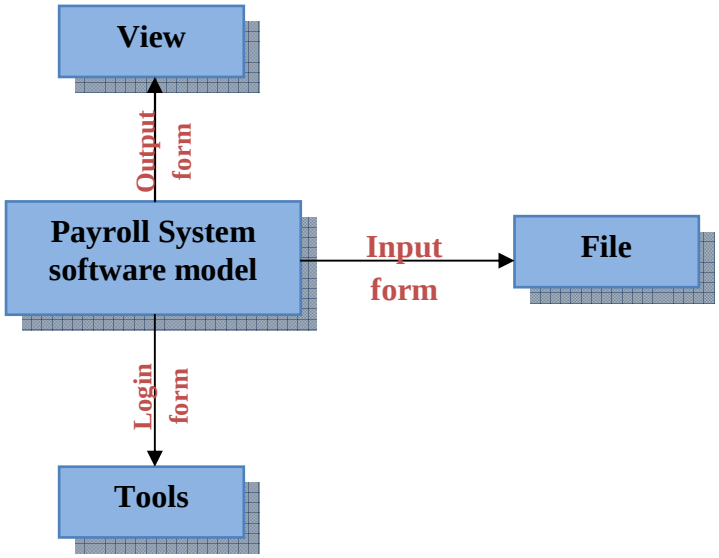

Fig 3: Payroll System software model

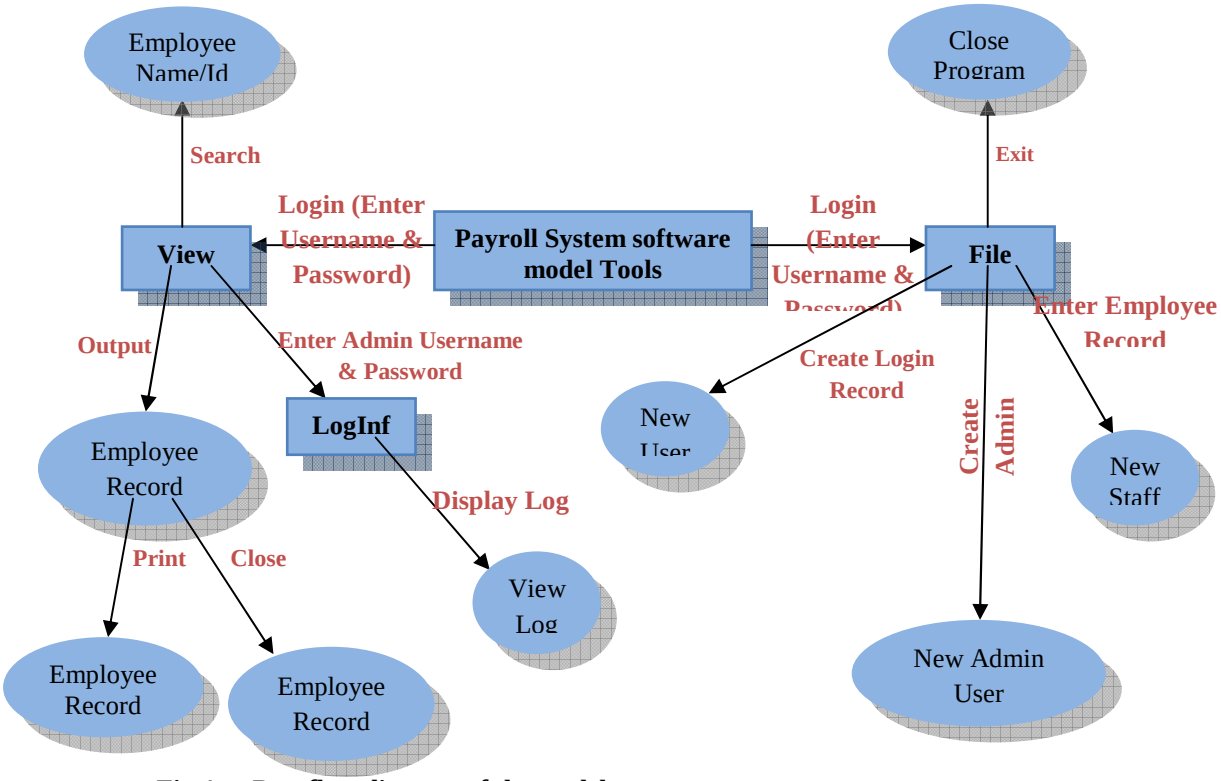

Fig 4: Dataflow diagram of the model

CONCLUSION

The potential exists for SOAP to allow one to set up very dynamic web services highly customized to the specific needs of each individual. However, this new opportunity comes with the challenge of being able to consistently provide flexibility without compromising security. Meeting the challenges of security requires knowing what the security needs and priorities are, what technologies can be used to achieve them, and above all, thinking clearly about your system's weaknesses [24].

A payroll system model was designed and implemented successfully. The model was to complement the efforts of the existing security requirements by adding an additional security to the audit record. This model can be implemented easily. There were no changes to the existing security requirements; rather it helps strengthen the existing security requirements of the system and helps in apprehending a criminal even when he/she succeeds in the criminal activity.

REFERENCES

[1] Bruce Schneier (2000) *Crypto-Gram Newsletter June 15, 2000.* Available online at: http://www.counterpane.com

[2] IBM Corporation (2005) *WebSphere Commercial Server.* Available online at: http://publib.boulder.ibm.com/infocenter/wchelp/v5r6m1/index.jsp?topic=/com.ibm.commerce.admin.doc/concepts/csesecuritymodel.htm.

[3] Jim Clune and Adam Kolawa (2002) *Security issues with SOAP*. Parasoft Corporation. Cross Talk, The Journal of Defence Software Engineering. 2031 South Myrtle Avenue Monrovia, CA 91016

[4] Nicholas Quaine (2007) *Soap Basics*. Available online at: http://www.soapuser.com/basics1.html.

[5] Satoshi Hada and Hiroshi Maruyama (2000) *Soap Security Extensions*. Tokyo Research Laboratory, IBM Research. Available online at: http://www.trl.ibm.com/projects/xml/soap/wp/wp.html.

[6] Sridevi Krishnan (2003) Security Model for Web Services. Available online at; http://www.topxml.com/securitymodelforwebservices.

[7] Sridhar Ravuthula (2008) *Web Services Applications and Security: Part 2*. Available online at: http://www.developer.com/services/article.php/1550461.