Security Issues in Web Services

Kuyoro Shade O.[†] **Ibikunle Frank**[††] **Awodele O.**[†] **and Okolie Samuel O.**[†]

[†]Babcock University, Ilishan-Remo, Ogun State, Nigeria [††]Covenant University Otta, Ogun State, Nigeria

Summary
Web Services are a promising solution to an age-old need: fast and flexible information sharing among people and businesses. They represent the next phase of distributed computing, building on the shoulders of the previous distributed models. Web Services leverage the ubiquity of the Internet to link applications, systems, and resources within and among enterprises to enable exciting, new business processes and relationships with customers, partners, and suppliers around the world. They enable access to data that has previously been locked within corporate networks and accessible only by using specialized software. Along with the benefits of Web Services comes a serious risk: sensitive and private data can be exposed to people who are not supposed to see it. The security issues of Web Services in a distributed environment are a major concern of research. Web Services will never attain their tremendous potential unless we learn how to manage the associated risks. The paper therefore focuses on the general framework of security issues and the proposed solution to web services security risks.

Key words:
security, web services, distributed computing, link applications.

1. Introduction

Web Services (Neil, 2003) are loosely coupled self-contained, self-describing and modular applications that can be described, published, located and invoked over a network. Web services can be provided on any platform and may be written in any programming language. Web services are the newest incarnation of middleware for distributed computing and unlike all previous forms of middleware, it is a simpler, standards-based, and more loosely coupled technology for connecting data, systems, and organizations. Web Services essentially involve the three roles of Service Oriented Architecture (SOA): service provider, service requester and service broker. A service provider could be an industry, business or a company capable of providing service. A requester also could be a company or a business that is in need of the service, where as the broker is a place, entity or a system that helps both service provider and service requester to discover each other. Basically, four technologies form the basis of Web services: eXtensible Markup Language (XML); Simple Object Access Protocol (SOAP); Web Services Description Language (WSDL); and Universal Description, Discovery, and Integration (UDDI).

XML: eXtensible Markup Language (XML) was created as a structured self-describing way to represent data that is totally independent of application, protocol, vocabulary, operating system, or even programming language. XML was initially developed to overcome the limitations of HTML, which is good at describing how things should be displayed but is poor at describing what data to be displayed.

SOAP: Simple Object Access Protocol (SOAP) is used for communication among different Web Services. SOAP was created as a way to transport XML from one computer to another via a number of standard transport protocols. HTTP is the most common and the most prevalent transport used by the Web itself. SOAP (Mcintosh and Austel, 2005) messages flow from originator to an ultimate receiver through a SOAP message path. A SOAP message consists of Soap Envelope which contains Soap Body element and an optional Soap Header element. The Soap Header element may contain a set of child elements that describe message processing that the sender expects a recipient to perform. Below is a typical SOAP listing.

```
01 <Soap: Envelope—-□
02 <Soap: Header (optional)>
03 <Soap: Body> (mandatory)
04 <get Quote symbol = "——"/>
05 </Soap: Body>
06 </Soap: Envelope>
```

Listing 1: A Simple SOAP message

SOAP envelope is used to encapsulate the SOAP message. SOAP header is the optional part of the SOAP protocol. Header contains information for the SOAP node, the processor of the SOAP message, how to process the SOAP message. This may be authentication, routing etc. Soap body contains the targeted to the SOAP message receiver. Get Quote element is the child of SOAP body.

WSDL: Web Service Description Language (WSDL) is used to describe the functionalities of the services. It is an XML language that defines what the input and output structure will be for a Web service, and what one expects to see in the payload XML message. WSDL is how one service tells another which way to interact with it, where the service resides, what the service can do, and how to invoke it. Once the requester receives the WSDL document for the candidate Web service, it must be validated. The simplest method of doing this is to provide a digital signature of the WSDL document for the requester to use. Requesters cannot connect to most providers without some form of authentication.

UDDI: Universal Description Discovery and Integration (UDDI) is used as a registry of information for Web

Manuscript received January 5, 2012
Manuscript revised January 20, 2012

Services, that is, to publish and discover information. UDDI service is an industry-wide effort to bring a common standard for business-to-business (B2B) integration. It defines a set of standard interfaces for accessing a database of Web services. The purpose of UDDI is to allow users to discover available Web services and interact with them dynamically. The process can be divided into three phases: Searching (discovery), Binding and Executing.

Web Service is an attractive and powerful technology for development of distributed application as well as for integration. Web Services provides interoperability across security policy domains. But, while they offer attractive advantages, Web Services also present daunting challenges relating to privacy and security. These range from random acts of Net vandalism to sophisticated, targeted acts of information theft, fraud, or sabotage. What makes security for Web Services so challenging is the distributed, heterogeneous nature of these services. For wide acceptability by the developers and consumers in business-to-business (B2B) and business-to-consumers (B2C) scenarios, Web services must be secured. Therefore study of security issues in Web Services is a need of the hour. This work presents what the security challenges of Web Services are and how to face them. The following section describes the security trend of web services. The remaining sections are arranged as follows. Section 3.0 presents general security frameworks of web services, section 4.0 describes various security threats to web services, section 5.0 presents current technologies in web services security, section 6.0 highlights the proposed solution to web services security risks and section 7.0 concluded the discussion with some future directions.

2. Security Trend of Web Services

Twenty years ago life was reasonably simple for the security professionals. Sensitive data resided on monolithic back-office data stores. There were only a few physical access paths to the data, which were protected by well-understood operating system access control mechanisms. Policies, procedures, and tools had been in place for many years to protect legacy data stores. Then, several years ago, Web-based applications came on scene with the advent of e-commerce such that secure access to Web servers was extremely important. Today, there are many mature perimeter security technologies, such as Secure Socket Layer (SSL), firewalls, and Web authentication/authorization servers that enforce security between browser clients and corporate Web servers.

Figure 1 illustrates new and existing security mechanisms for securing Web Services at different security tiers.

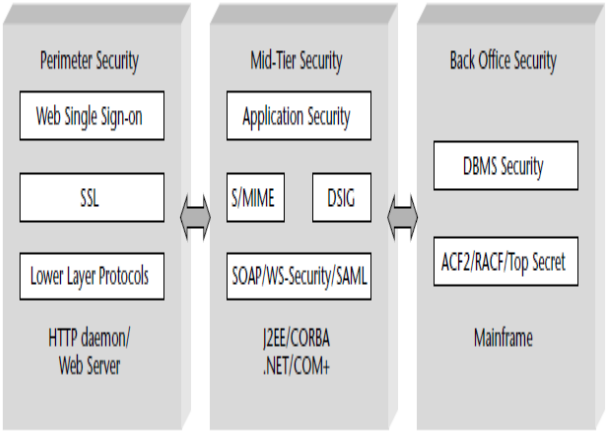

Fig. 1 A typical Web Service Security implementation

3. General Security Framework

Some core security services that are fundamental to end-to-end application security across multitier applications are defined here. They are:

Authentication: verifies that principals (human users, registered system entities, and components) are who they claim to be. The result of authentication is a set of credentials, which describes the attributes (identity, role, group, and clearance) that may be associated with the authenticated principal.

Authorization: grants permission for principals to access resources, providing the basis for access control, which enforces restrictions of access to prevent unauthorized use. Access controls ensure that only authorized principals may modify resources and that resource contents are disclosed only to authorize principals.

Cryptography: provides cryptographic algorithms and protocols for protecting data and messages from disclosure or modification. Encryption provides confidentiality by encoding data into an unintelligible form with a reversible algorithm, which allows the holder of the decryption key(s) to decode the encrypted data. A digital signature provides integrity by applying cryptography to ensure that data is authentic and has not been modified during storage or transmission.

Availability: states that resources, services should be available to authorized parties at all times.

Accountability: ensures that principals are accountable for their actions. Security auditing provides a record of security-relevant events and permits the monitoring of a

principal's actions in a system. Non-repudiation provides irrefutable proof of data origin or receipt.

Security administration: defines the security policy maintenance life cycle embodied in user profiles, authentication, authorization, and accountability mechanisms as well as other data relevant to the security framework.

Integrity: asserts that no one has tampered with a message since it was initially created. This assures the sender and the receiver that every bit produced by the sender is received by the recipient in precisely unaltered form. Data integrity is accomplished by using digital signatures. Messages in which data integrity is required must explicitly or implicitly include the identity and credentials of the sender to enable this kind of message-level security.

Confidentiality: keeps the message secret. This process requires encryption, which scrambles the message in such a way that only authorized identities can decrypt and see the data. To do this, a shared secret and an algorithm for encrypting and decrypting the message is exchanged. In the real world, these algorithms are very challenging mathematical functions with keys that are very large numbers, and the time to do the analysis is technically infeasible even with modern computers.

Non-repudiation: proves that one identity sent the data only to another identity. This then proves that the specific transaction was entered into by the recipient, and neither party can refute or deny that it occurred later. If the transaction is challenged legally, a contract that was supposedly executed must be shown to have been entered into by both parties. Each party must have seen the contract signed, and their identities -confirmed traditionally by validating wet signatures on paper and notary witnesses- must have been confirmed at the time of signing. These are difficult, and as yet legally unchallenged, tenants to uphold in a digital and anonymous world, but that day is coming. Non-repudiation depends on public key cryptography technology.

4. Security Threats to Web Services

There are many complexities specific to, and inherent in Web services that complicate their security. Numerous threats can compromise the confidentiality, integrity, or availability of a Web service or the back-end systems that a Web service might expose. Some of these threats are shared with conventional Web application systems (Web sites), while others are specific to Web services. The following are the general security threats that can occur in any Web application.

SQL Injections: When SQL statements are dynamically created as software executes, there is an opportunity for a security breach. If the hacker is able to break perimeter security and pass fixed inputs into the SQL statement, then these inputs can become part of the SQL statement. SQL injections can be generated by inserting spatial values or characters into SOAP requests, Web form submissions, or URL parameters. A hacker who knows his SQL can use this technique to gain access to privileged data, log-in to password-protected areas without a proper log-in, remove database tables, add new entries to the database, or even log-in to an application with admin privileges.

Capture and Replay Attacks: As Web messages are transmitted over the Internet, they are prone to man-in-the-middle attacks. Such an attack occurs when a malicious party gains access to some point between the peers in a message exchange. For instance, a hacker might capture and replay a SOAP request to make a monetary transfer, or modify the request before it reaches its destination - ultimately causing severe losses for any of the peers in the message exchange.

Buffer Overflows: Native applications can suffer from unchecked input data sizes. If inputs are not validated, a buffer overflow attack can transpire remotely via SOAP requests or Web form submissions. Buffer overflow attacks occur when a hacker manages to specify more data into one or more fields and write to the buffer beyond the size of the memory allocated to hold the data. Buffer overflows can result in application or system crashes or, when crafted carefully, they can even allow attackers to compromise the system and access unauthorized information or initiate unauthorized processes. The hacker can exploit this weakness so that the function returns to a hacker-designated function, or so that the function executes a hacker designated procedure.

Denial-of-Service Attacks: Denial-of-service (DoS) attacks are launched to compromise system availability. There are two ways to mount DoS attacks. First, attackers can consume Web application resources to a point where other legitimate users can no longer access or use the application. This can be accomplished by sending a query for large amounts of data. The second approach can occur when attackers lock users out of their accounts or even cause the entire application to fail by overloading the service with a large number of requests. Attackers could combine these two approaches with Web service specific-attacks to maximize damage.

Improper Error Handling: Many application servers return details if an internal error occurred. Such details typically include a stack trace. These details are useful during development and debugging, but once the application is deployed, it is important that such details do not find their way to regular users because the details may include information about the implementation and could expose vulnerabilities. For instance, an error message about a bad SQL query indicates to a malicious user that

his or her inputs are used to generate database queries, thus possibly exposing SQL injection vulnerability. Another instance, a request that includes a wrong username or password should not be met with a response that indicates whether or not the username is valid; this would make it easier for an attacker to identify valid usernames, and then use them to guess the passwords.

Eavesdropping: Eavesdropping is another security risk posed to web services. Classified information and transactions are frequently transmitted using Web services. By carefully examining the data, attackers can eavesdrop to intercept SOAP messages and read all of the information contained therein. Therefore, it is important to maintain a secure transmission so that this type of eavesdropping by unauthorized parties is eliminated. Some of the most damaging things that get sniffed include passwords and credit card information.

Session Hijacking: Session hijacking involves gaining illegal control of a legal user's session state. It occurs when an attacker steals a valid session ID (valid session cookie), and uses it to gain that particular user's privileges in the application. By intercepting or sniffing SOAP messages, an attacker can hijack a user's session in the same ways as with normal web application attacks, however once a hacker is authenticated as a valid user he may perform more dangerous activities.

5. Web Services Security Current Technology

WS-Security: WS-Security is a building block that is intended to be used in conjunction with other Web Services and application specific protocols; to accommodate a wide variety of security models. WS-Security (Kearney et al., 2004) does not claim to provide a complete solution to securing Web services. The XML signature and XML encryption specifications provide standard methods for digitally signing and encrypting XML documents including SOAP messages. Not only can whole documents be signed or encrypted, but also individual parts. WS-Security defines how XML signature data can be included in a SOAP message. This provides persistent confidentiality beyond a single SOAP communication.

Secure Socket Layer: Secure Socket Layer (SSL) is a protocol or technology, which is used to protect companies from Web Service Security attacks. SSL used in encryption technique, which are in turn used to implement for data protection. SSL creates a secure tunnel in between originator and destination computers based on public key encryption technique. A common protective measure is to send messages over a secure connection that is using SSL. For instance, an SSL connection between two points may be sufficient for simple applications. For multiple Web Services, complete message or individual part of messages may be encrypted and signed to protect the confidentiality and integrity of Web Service messages (Kearney et al., 2004).

XML Encryption: XML Encryption provides end-to-end security for applications that require secure change of structured data. XML Encryption is mainly ensuring confidentiality to encrypt the XML data. XML based Encryption is the natural way to handle requirements for security in data interchange applications. XML Encryption is not intended to replace or supersede Secure Socket Layer (SSL). Rather, it provides a mechanism for security requirements that are not covered by SSL. XML encryption is ideal for confidentiality. XML Encryption does not introduce any new cryptography algorithms or techniques. RSA Encryption may still be used for actual encryption.

SAML: Security Assertion Markup Language (SAML) is a protocol for asserting authentication and authorization information. It also provides attributes of an end-user in XML format. It allows information to be placed on a SOAP message. SAML servers can be accessed for authentication and authorization data in order to enable Single- Sign-On (SSO). If the recipient of this SOAP message trusts the sender of the SAML data, the end user can also be authorized for the Web Service.

XACML: eXtensible Access Control Markup Language or XML-Access Control Markup
Language (XACML) is designed to express access control rules in XML format. Although the two technologies are not explicitly linked, XACML may be used in conjunction with SAML. An authorization decision expressed in a SAML assertion may have been based on rules expressed in XACML.

6. Proposed Solution

The main objective of this work is to bring into focus the review of security issues in Web services and discuss a common framework of general security issues. Also some new security issues in Web Service Security (WSS) along with their attacks are highlighted. The key to effective Web services security is to know the threats as described above, understand the technical solutions for mitigating these threats, and then establish and follow a defined engineering process that takes security into consideration from the beginning and throughout the Web service life cycle. This process can be established in the following four steps:

1. Determine a suitable Web service security architecture.
2. Adhere to technology standards.

3. Establish an effective Web services testing process.
4. Create and maintain reusable, re-runnable tests.

By following these four steps, one can ensure complete Web service security.

7. Conclusion

Web services security is still a work in progress and one needs to understand the potential security risks and proactively minimize those risks so that Web services are less vulnerable to attack. In the present work a review of security issues in Web services is done under a common security framework. Some new security issues are highlighted. New security architecture based upon Web services that support authentication, authorization and integrity is a central point for future research.

References

[1] Cerami E. (2002). Web Services Essentials: Distributed Applications with XML-RPC, SOAP, UDDI & WSDL. O'Reilly Publisher

[2] Hartman B., Flinn D. J., Beznosov K. and Kawamoto S. (2003). *Mastering Web Services Security*. Wiley Publishing Inc. Indianapolis, Indiana

[3] Kearney P., Chapman J., Edwards N., Gifford M. and He I. (2004). An Overview of Web Services Security. *BT tech. Journal*. 22(1): 27-42

[4] Mcintosh, M.; Austel, P.(2005). XML Signature Element Wrapping Attacks and Countermeasures. Fairfax. Virginia. USA

[5] Michael N. H.; Singh, M. P.(2005). Service-Oriented Computing: Key concepts and principles. *IEEE Internet computing*. 9(1):75-81

[6] Neil. M.O. (2003). *Web-Service Security*. Tata Mcgraw-Hill Pub. New York

[7] Peterson G. and Lipson H. (2006). Security Concepts, Challenges, and Design Considerations for Web Services Integration. Carnegie Mellon University and Cigital, Inc

[8] Rami Jaamour (2005). Securing Web Services. Information System Security www.infosectoday.com

[9] Sinha S. , Sinha, S. K. and Purkayastha B. S. (2010). Security Issues in Web Services: A Review and Development Approach of Research Agenda. *Assam University Journal of Science & Technology: Physical Sciences and Technology*. 5(2):134-140

Kuyoro Shade O. received the B.Sc. and M.Sc. degrees in Computer Science from Olabisi Onabanjo University (2004) and University of Ibadan (2010) respectively. She is an assistant lecturer at Computer Science Department, Babcock University, Ogun State, Nigeria. Her research interests are in the area of computer networking, machine learning and artificial intelligence.

Ibikunle Frank is a Senior Lecturer at Computer Science Department, Covenant University Otta, Ogun State, Nigeria His research interest is in the area of computer networking.

Awodele Oludele is a Senior Lecturer at Computer Science Department, Babcock University Ilishan-Remo, Ogun State, Nigeria. His research interest is in the area of artificial intelligence.

Okolie Samuel O. is a Senior Lecturer at Computer Science Department, Babcock University Ilishan-Remo, Ogun State, Nigeria. His research interest is in the area of numerical analysis.