Identifying SOA Security Threats using Web Mining

Mohamed Ibrahim B
Software Solution Architect & Research Scholar

Mohamed Shanavas A R
Associate Professor, Jamal Mohamed College, Tiruchirappalli, India 620 020

ABSTRACT
The Service Oriented Architecture (SOA) became a dominant paradigm for enterprise computing. The web services are the implementation of SOA that works for heterogeneous platforms as they use common Internet protocols for communication and simple text format such as XML for data representation. The basic SOA architecture does not contain any security solution within it and the security is applied to SOA as an ad-hoc manner and also it depends on the internal architecture of the security products. There is no comprehensive security solution is achieved yet for SOA. This paper identifies the SOA security threats using the data mining technique—web mining. At the end of this paper, a customizable security solution for SOA in the form of framework is presented.

General Terms
Security, Service Oriented Computing, Web Services, Algorithms

Keywords
SOA Security, Web Services, SOAP, Data Mining, Web Mining, WSDL, REFTful Web Services

1. INTRODUCTION
The Service Oriented Architecture (SOA) is the emerging paradigm in the current era of enterprise computing to achieve Enterprise Application Integration (EAI). Many different solutions in terms of architectures, prototypes, models and framework were proposed as open source and vendor products for integrating interrelated applications in an enterprise for form EAI. However, they are not able to achieve any landmark in the construction of a comprehensive security solution that works for heterogeneous platforms, as these solutions are dependable to the internal architecture of the products and they are developed for specific platforms.

The Web Service is the form of SOA implementation that works for cross-platforms as it uses common Internet protocols such as HTTP. The service is a Software component that fulfills a specific functionality and it does not depend on the context of other services. Thus, SOA exposes Software resources in the form of services, which can be accessed over network [1]. According to [2], the Service Oriented Architecture is defined as an open agile, extensible and federated architecture comprised of autonomous QoS capable, vendor diverse, interoperable, discoverable and potentially reusable services implemented as Web-Service for organizing and utilizing distributed capabilities that may be under the control of different ownership domains.

The basic architecture of SOA consists of three main components: (i) Service Provider, (ii) Service Registry, and (iii) Service Requester [3] as shown in Figure-1. The Service Provider is the entity that owns web services running and publishes its description to the Service Registry, where the Service Requester finds the required service description from the Service Registry and initiates communication with Service Provider. The Web Services Description Language (WSDL) is used as the standard for describing the web services information and publishing into Service Registry. The Simple Object Access Protocol (SOAP) is the standard that uses XML format for request/response between Service Requester and Service Provider. The three core operations that can be performed in basic SOA architecture are: (i) Publish, (ii) Find, and (iii) Bind [4].

The WSDL is an XML document designed according to the standards specified by the World Wide Web Consortium (W3C) that describes exactly how a specific web service works [5]. The UDDI directory is used as a registry of web services that are available for use in a particular network. It would tell us where to find the service, also to examine the WSDL document. The SOAP specification provides standards for the format of a SOAP message and how SOAP should be used over HTTP. SOAP has been created to transport XML documents from one computer to another and it can be used with a number of standard transport protocols; it is the binding part of web services [6].

Fig 1: Basic SOA Architecture

The following mining techniques [7] are applied in the identification process of SOA security threats.

Web Mining:
- Predictive Modeling
- Clustering
- Association
- Deviation Analysis

Process Mining:
- Discovery (defining process model)
- Conformance (used to check if reality confirms to the model)
extension to enrich the model

Web Usage Mining:
- Path and Usage Pattern Discovery
- Pattern Analysis

The remaining sections of this paper are described as: Section 2 explains the security requirements for SOA. Section 3 reviews the available literature for SOA security, the security for REST web services are outlined in Section 4, the identified SOA security threats are briefed in Section 5, the proposed SOA security framework is outlined in Section 6, and Section 7 concludes the paper.

2. SECURITY REQUIREMENTS FOR SOA

Though the fundamental technology of Web Services, XML (eXtensible Markup Language), has given provided Web Services with many advantages but unfortunately it also caused many problems in security concerns too [8]. Web Service infrastructures introduce new threats to web-based applications as well as new challenges when it comes to securing them [9].

According to [10], a threat is a possible way a system can be attacked and the threats can be broadly categorized in four classes according to their consequences: (i) Disclosure – it is the unauthorized access to data, (ii) Deception – it is the provision of false data which is believed to be true, (iii) Disruption – it aims at preventing an asset from correct operation, and (iv) Usurpation – it leads to losing control of the asset to an unauthorized entity.

One of the major design issues of SOA is meeting its security requirements, since it affects interaction of services and applications in SOA environment [11]. Security must be unified with the software engineering process [12] and security engineering [13] should be implemented for SOA applications at design phase itself, instead of applying as an ad-hoc requirement. Successfully implemented SOA security has to be well-defined, well-priced, and well-implemented [14].

The threats on SOAP based Web Services include, Message alteration, Loss of confidentiality, Falsified messages, Man in the middle, Forged claims, Capture-reply of message, Replay of message parts, Denial of services, XML external entity attacks, XPath/Field/SQL injection, Harmful SOAP attachments, XML dereference attacks, XML recursion attacks, XML document size attacks, XML flooding, Dictionary attacks, Cookie poisoning, Data tampering, Message spoofing, WSDL enumeration, Routing detour, Schema poisoning, Malicious morphing, Memory barrier breach, XML virus, Buffer overflows, Recursive elements, Resource hijacking, Cross site scripting, Eavesdropping, Spamming, IP spoofing, Phishing, Pharming, Malicious programs and Malicious file execution, Worms, Rootkits, Botnets, Identity theft, XML parser attacks, Jumbo payloads, and many more.

3. SOA SECURITY IN THE VIEW OF LITERATURE

Transport layer security mechanisms such as TLS or SSL are commonly used to provide data integrity and confidentiality between two communicating entities. The problem arises when the concept of intermediary nodes are introduced. The point-to-point principle implies that a message has to be decrypted at the intermediary node and then re-encrypted before it is forwarded to the final destination. This model cannot guarantee that the intermediary service does not manipulate the data either intentionally or unintentionally, thus end-to-end security is not ensured [15].

The available security solutions work only for trusted parties on secured channel. The service requests from anonymous customers may lead to security vulnerabilities and these security threats are not addressed in the existing solutions [16]. In the same way, digital signatures alone do not provide message authentication as the attackers can record and resend signed message. Unfortunately, XML Encryption and XML Signature do not solve all web services security problems, mainly due to the data are available in plain text format.

Bhavani Thuraisingham [17] states in her book while talking about Secure Services that the Web services and Service Oriented Architectures are at the heart of the next-generation web. However, one of the major problems in Service Oriented Architectures is ensuring a secure infrastructure environment [18] as service communications are made by machine to machine. With the introduction of Web 2.0, Web 3.0 and second generation internet based services, the traditional security approaches such as Secure Socket Layer / Transport Layer Service (SSL/TLS), and Virtual Private Network (VPN) become obsolete. Hence the organization should extend SOA security framework to adopt with these new generation technologies [19, 20].

4. SECURITY FOR REST WEB SERVICES

The Representational State Transfer (REST) is an architecture design for the networked applications, where the requests and responses are built around the transfer of representations of resources. Each resource has one or more representation such as XML and JSON. RESTful web service is a web API that is implemented using HTTP and the principles of REST.

In the REST architecture, data and functionality are considered resources, and these resources are accessed using Uniform Resource Identifiers (URIs), typically links on the web. The resources are acted upon by using a set of simple, well-defined operations. The REST architecture is fundamentally client-server architecture, and is designed to use a stateless communication protocol, typically HTTP. In the REST architecture, clients and servers exchange representations of resources using a standardized interface and protocol.

RESTful web services are web applications built upon the REST architecture. They expose resources (data and functionality) through web URIs, and use the four main HTTP methods POST, GET, PUT, and DELETE to create, retrieve, update, and delete resources respectively. RESTful web services typically map these four main HTTP methods to the so-called CRUD actions: create, retrieve, update, and delete.

It is possible for SOAP web services to provide end-to-end security even the SOAP messages pass-through the intermediaries as required by the underlying network. But with RESTful web services, only message security is provided by transport protocol (HTTPS) which offers only point-to-point security. Also it is not possible to re-encrypt the messages in case of any communication failure.

Thus REST is a modern architectural design for connecting networked applications. The core idea is to utilize the simple HTTP protocol instead of complex mechanism such as RPC.
CORBA, and even SOAP [21,22], RESTful web applications typically have the same attack vectors as standard web applications, including injection attacks, XSS cross-site scripting, broken authentication and cross-site request forgery (CSRF) [23].

The RESTful web service uses session based authentication either by establishing a session token or using a cookie. However, the usernames/passwords and session tokens should not appear in the URL query string, as this can be captured in web server logs and makes them intrinsically valuable [24]. To overcome replay attacks, REST service providers are encouraged to use time limited encryption key, encrypted session token and incoming IP address. Protection against using HTTP methods is also mandatory as the user can access POST (create) / PUT (update) methods also using the user’s privilege where the service provider is expecting the user to access only GET (read).

The following are some of the proposed security implementation for RESTful web services [25,26].

- Input validation
- Secure session tokens
- Privileged HTTP method access
- Secure direct object references
- Secure parsing of input requests
- Strong typing of data
- Validate incoming content types (MIME)
- Validate responses
- Secure HTTP headers
- Apply XML encoding and JSON encoding
- Encrypt messages
- Apply/use TLS (HTTPS), HTTP Digest, Two/Three-legged OAuth algorithms to ensure secure transmission of data and integrity.

5. SECURITY THREATS ON SOA

The following sub-sections briefly explain some of SOA security threats and attacks that are identified through web mining techniques applied on the real-time web service architecture.

5.1 Threats on Identity of data

These are authentication and authorization threats that the intruder aims on ‘identity’ data.

5.1.1 Dictionary Attacks

The attacker’s goal is to obtain passwords. An attacker systematically tests all possible passwords to perform dictionary attack. To perform dictionary attack means trying every word in the dictionary until matching password is found. Most password-based authentication algorithms are vulnerable to dictionary attacks [27].

5.1.2 Dictionary Attacks

These attacks use the raw computer processing power to try different permutations of any variable that could expose a security hole. For example, if an attacker knew that access required an 8-character username and a 10-character password, the attacker could iterate through every possible combination in order to attempt to gain access to a system. No intelligence is used to shape or filter likely combinations. [28]

5.1.3 IP Spoofing

Performing IP-spoofing attack, an attacker fakes IP address to deceive receiver to believe it is sent from a location that it is not actually from [29].

5.1.4 Message Eavesdropping

Performing Network Eavesdropping attack, the attacker can capture traffic and obtain sensitive information such as usernames and passwords. The data to a web method that are passed in plaintext format from client to server allow the attacker to collect sensitive data by eavesdropping the connection between these two nodes. Through Message Eavesdropping attack, the attacker gathers sensitive or other information through analyzing Web Services messages. The victim usually knows nothing of the attacker’s presence, or that he/she, i.e. victim is being monitored [30].

5.1.5 Data Tampering

Data tampering occurs when an attacker changes or modifies legitimate data with illegal data, while it passes over the network [31].

5.2 Threats on Session of WS Transactions

Web service provider establishes session in communication with service requester as a web service application request and response. Performing session attacks the attackers may capture messages or insert false instructions.

5.2.1 Replay Attacks

The attacker captures input messages, and sends it repetitively, which results in overloading of Web Service [32]. This attack is similar to the “network ping of death” in which a hacker can issue repetitive SOAP message requests in a bid to overload a Web service. The attacker bypasses authentication by sniffing URL parameter or Cookie value; the attacker can replay that hijacked token to the application. Hence by replaying a request, the attacker can gain access to the system under a false identity, i.e. under identity of legitimate user [31].

5.2.2 Man-in-the-middle Attacks

Man-in-the-middle Attacks Since SOAP message passes through multiple intermediate systems, there is a possibility of attack in the intermediary systems. For example, compromising one of intermediate station opens the possibilities for attackers to perform man-in-the-middle attack by inserting fake/bluff routing instructions so that message travels to malicious location and from there the attacker can send malicious instruction to the original destination [33]. It is also possible by an attacker to intercept messages between two hosts i.e. client and server. Both client and server assume that they are communicating with each other, but the communication between the sender and recipient is actually flowing through the attacker. The attacker is free to modify the content of the messages and sent them to the original recipient [34].

5.2.3 Session Management Attacks

During the Session Management attacks, the attacker is able to manipulate a session token, which can be a cookie, URL parameter, session ID, or any obfuscated value that is passed between the application and browser. Analyzing tokens gives the attackers an opportunity to identify vulnerabilities in the application [8, 35]
5.3 Threats on Parsing data
There are possibilities of attacks at parsing XML, where Web Services input and output data are exchanged in XML format.

5.3.1 Recursive Payloads
The attacker attempts to stress and break an XML parser by sending mass amounts of nested data [32]. Normally XML elements are nested for representing group of data. But in this attack, the attacker do nesting of elements to higher number of deep, say 100,000 so that the XML Parser will be stressed and even will be broken its operation while processing these deeply nested elements.

5.3.2 Oversized Payloads
The attacker attempts to overload the parser by sending unlimited file size, in order to make Denial-of-Service [32]. For example, oversized payloads of XML could cause a DOM based parsers to malfunction as DOM (Document Object Model) based XML parsers load the entire XML document into main memory before parsing.

5.3.3 Schema Poisoning
XML Schemas provide formatting instructions for parsers when interpreting XML documents. Attackers compromise XML Schema and replaces it with similar, but a modified one, resulting schema poisoning. This attack may lead to XML Denial-of-Service attacks [29, 36].

5.4 Threats on Web Service Code
Code attacks intended to affect applications that run Web Services, for example malicious code carried by XML messages that pass through the web server port. Successfully performed code attacks give the opportunity for attackers, for example to extract the whole XML database.

5.4.1 SQL Injection
SQL Injection attack occurs when malicious SQL statements are inserted into XML in order to disrupt the back-end system, for example trying to receive data that it is not authorized to access, or even destroy the data [37].

5.4.2 XPath Injection
An XML document has no access control or privilege system associated with it. By performing XPath injection there are possibilities for attackers to extract the whole XML database [38].

5.4.3 Cross-site Scripting
Cross Site Scripting (also known as XSS or CSS) vulnerability is caused by the failure of a site to properly validate user input before returning it to the client’s web browser. The attacker inspects applications and chooses the one that does not filter the input and at which the user is authenticated. The attacker inserts a malicious code in the request, and this will be returned to the victim by that application. The malicious script runs with the privileges of a legitimate script originating from the legitimate web server. The browser does not know that the code is not legitimate, because the script code is downloaded from a trustworthy site. Example: Trojan horse attacks [39].

5.4.4 Improper Error Handling Code
Many application servers return details if an internal error is occurred. Such details typically include a stack trace. These details are useful during development and debugging, but the attacker can guess the written code if these verbose debugging information is sent as response to a web service call. For example, if the debugging information outputs the username is valid and the password is wrong for a typical login process, then the attacker can confirm on the username and he/she will continue in breaking password for that particular user.

5.5 Threats on WSDL
Performing WSDL attacks the attackers analyze and tamper WSDL information and tamper with parameters within WSDL documents [40]. It is apparent that, WSDL document includes all of the operations that are available to the consumers, it is straightforward for a hacker to run through all of the operations with different message request patterns until a breach is identified.

5.5.1 WSDL Scanning
A WSDL document contains information such as the list of web-methods, the parameters for those methods, and types of I/O [41]. Lindstrom [32] points out that through scanning the WSDL document an attacker may reveal sensitive information like types, messages, operations, port types, bindings, and guess other methods.

5.5.2 Parameter Tampering
According to Lindstrom [32], attackers tamper parameters in order to retrieve unauthorized information. If so, attacker can inject malicious code into XML parameters. One of the aims of parameter tampering attacks is to modify parameters sent between the user and the application.

5.5.3 XML Wrapping
If the attackers successfully get the WSDL file, they may exploit XML wrapping to bypass authentication. XML wrapping works more or less in the same way as SQL injection. Because of the message send by user is in XML format, all the input data will be wrapped into XML tags, called elements. Consequently, if the user input some elements, the SOAP message will treat them as its inherent elements as well. Under this condition, hackers can modify unauthorized data by injecting relevant elements [42].

5.6 Overflow Attacks
This kind of attacks is aimed at the service endpoint and SOAP engine through Web server. The attackers send parameters longer than the program can handle, which could cause the service to crash.

5.6.1 Buffer Overflow Attacks
When performing a buffer overflow attack, an attacker put a larger amount of data than expected into program variable. The amount of memory, reserved for the operation becomes smaller than the amount of data written to the memory. As result the extra data goes somewhere undesirable. Poorly written code, such as a program that inserts data into a buffer and does not check the size of the data being inserted, often invites buffer overflow attacks [43].

5.7 DoS Attacks
In Denial of Service (DoS) attack, the attacker’s main goal is to reveal information that he/she can use for crashing the Web Service server. Performing DoS the attacker may attack a router, firewall, or proxy server with the goal of making them unusable. By attacking proxy server, for example, an attacker can redirect his malicious traffic for own benefit.

5.7.1 XML Denial-of-Service (XDoS) Attack
An attacker tries to prevent legitimate users from accessing a service by flooding the service with thousands of requests, as a result, the server receives more requests than it can handle. The effect of this DoS attack is to prevent the service-
providing server from being able to provide the service [44, 45].

5.8 UDDI Vulnerabilities
Web Service requesters can query the Universal Business Registry (UBR) to determine the details of the services for them to use. From UDDI, the web service client can discover information about the service name, services provided by the web service publisher and the details of the WSDL file used to invoke the service as well as the location of the service. The attacker can utilize the information in order to perform attacks.

5.9 Attacks based on Reconnaissance
The attackers can investigate and collect information from openly available sources like WHOIS databases and DNS servers [46]. This information enables the attackers to try to access unauthorized web services.

5.10 HTTP Header Manipulation
HTTP header consists of control information that is passed between the client and server. The attacker can write his/her own program to manipulate the HTTP headers by handling requests which results the target service is attacked.

5.11 Threats on Broken Access Control
This occurs when restrictions on what authenticated users are authorized to do are not properly enforced. For example, if there is no proper access control is defined, an attacker can try to access other web service methods of the same web service or different web service running on the same web server using the login privileges given for accessing certain web methods.

5.12 Malicious SOAP Attachments
Web services that accept attachments can be used by attackers to inject malicious content into the web server. The attachments that are delivered to a Web service can then be processed by other applications that have vulnerabilities.

6. PROPOSED SOA SECURITY FRAMEWORK
In order to enable service-based cross-organizational collaboration, the security of the participating systems, exchanged messages, and used communication channels has to be ensured. Achieving and guaranteeing basic IT security goals such as confidentiality, authentication, authorization, non-repudiation, integrity, and availability is an absolute must in this context and still an active and core topic both in research and industry. Although security introduces additional costs and has an impact on the Quality of Service, unsecured business transactions are not an option in most scenarios [47]. Security is one of the major concerns of SOA-based implementations, especially when it spans outside the boundaries of enterprise [1].

The success or failure of any non-trivial system depends heavily on its overall representation commonly known as Software Architecture [48]. Software architecture of a system has shown to be very important in the realization of system wide qualities such as security, performance, etc. [49]. Security, as a factor, should be applied at the architectural level.

It is not possible to provide total or absolute security. However, it is possible to identify the attackers’ move in advance for each individual SOA applications and this gives an opportunity for security experts to design and implement security for those SOA applications. This can be done with the help of web mining on web services.

A custom security Software element named “Intelligent Security Engine” (ISE) is introduced at the service provider end in the proposed SOA security framework. The ‘Intelligent’ part of ISE is configured through supervised Artificial Neural Networks (ANNs) learning techniques based on the organizational security policies. This ISE component is enough to cater WSDL and message level threats. The network learnt the security in such a way that if any malfunction is identified, then it will be reported to Rule Based Engine (RBE) for preliminary analysis of the threat and then it will be forwarded to the configured Centralized Decision Engine (CDE) which determines the kind of fault, then the Fraud Detection System (FDE) will be updated through the Knowledge Acquisition Process (KAP) of the established Artificial Neural Network (ANN). All these sub-components are interconnected within ISE. Thus ISE is capable to refuse service to any unauthorized service requester at any point of time.

The Proof-of-Concept (POC) application for the implementation and testing of the proposed framework is developed using Java technologies for a large-scale logistic domain. Instead of applying the security checks on the corresponding code inside at multiple places, the Aspect Oriented Programming (AOP) techniques are applied which enable us to separate out the security codes and apply on the concerns. The word ‘Aspect’ in Software development refers to the ‘unit of modularity’ of any cross-cutting concerns within the applications. The AOP is modularity based application development programming technique, where the modules are aspects. AOP is distinguished from Object-oriented Programming (OOP) as OOP is based on classes where AOP is based on aspects. In this way, the proposed security solution looks like a wrapper to the existing web service application code.

7. CONCLUSION
Security is one of the major concerns of SOA-based implementations, especially when it spans outside the boundaries of enterprise. The basic SOA framework does not possess any security and the available implementation of SOA security depends on the respective proprietor of the framework/implementation.

Apart from the standard SSL security for RESTful web services, any encryption standards and authentication algorithms such as OAuth are introduced in the industry. There are also message based end-to-end security mechanisms, like S/MIME, are practiced now a days. The TLS secure sessions and the specific and keys are applied on the fly, and the content will be encrypted again and again in the secure tunnel. RESTful web services threats include session attacks, parser attacks, SQL/code/LDAP injections, Deniel-of-service (DoS), malware, buffer overflow and replay attacks. This paper discussed the important limitations and delimitations of RESTful web services.

In this paper, the authors critically analyzed the severe SOA security threats that are possible on SOAP based and RESTful web services that are identified using web mining techniques. A SOA security framework is outlined with its architecture. The further study of the research includes identification of the severity of these identified SOA security threats.
8. REFERENCES


[34] Fitzgerald, J. Dennis, A. “Business Data Communications and Networking,” John Wiley and Sons, 2002


[38] Nuno Antunes, Nuno Laranjeiro, Marco Vieira, and Henrique Madeira, “Effective Detection of SQL/XPath Injection Vulnerabilities in Web Services,” IEEE International Conference on Services Computing (SCC ’09), 2009


