# Design and Implementation of Secure and Efficient Access Control Framework for SOA

A. Laxmikanth
Research Scholar
Rayalaseema University
Kurnool, AP.
hellokanth@gmail.com

Dr. M.Raja
Professor
Department of CSE
SICET-JNTUH,
Hyderabad.
rajadrm@gmail.com

**Abstract:** The Service Oriented Architecture (SOA) became a dominant model for enterprise computing. The web services are the applications of SOA that works for diversified platforms. Web services use common Internet protocols for communication and simple text format such as XML for data representation. The SOA fundamental architecture does not contain any security resolution. Therefore security is applied to SOA as an improvised manner and also it depends on the internal architecture of the security products. In addition to this, the data stored by web application in persistent storage needs to be encrypted so as to avoid any data exposure or chance of hacking. Data encryption is an efficient method of guaranteeing information security. There are many encryption methodologies available to encrypt the data.

**Keywords**. SOA, Authorization & Security Service, Authorization Service and Intelligent Data Mining in SOA

## I. Introduction

SOA is a collection of loosely coupled and independent services (or resources) each with a well-defined interface that help the service interacts with other services regardless of their implementation or platform. Services are offered on demand and can range from a simple service were only one service is involved to a higher level service composed of many services. Services can be delivered to an end user, application or to another service there is no need for human intervention. Services in SOA networks are most likely implemented using web services technology. This does not mean that other types of technologies are not applicable [1]. Any technology that promotes sharing, reuse, interoperability and has a mean for advertising the services is a technology that can implement SOA.

The most popular data format and protocol in use for web services are XML and SOAP [2]. In most SOA implementations a directory system known as UDDI is used for web service discovery and publication. Web servers advertise services using a well-defined interface or WSDL file and uses SOAP messages as a communication mechanism between web services.

The SOA fundamental architecture consists of three main components:

1. Service Provider
2. Service Registry
3. Service Requester

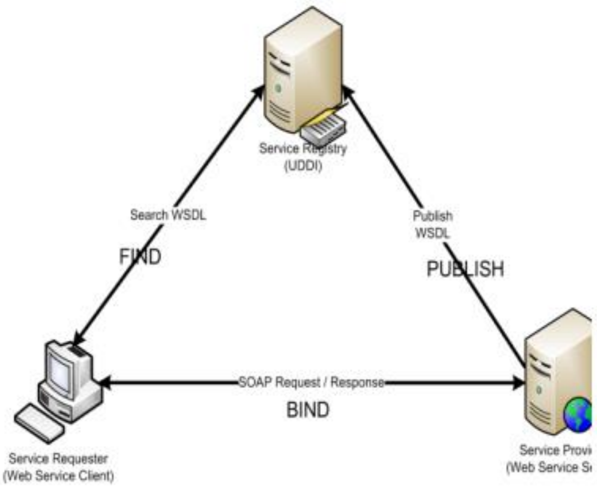


Fig 1: Basic SOA Architecture

The Service Provider is the component that owns web services running and publishing its description to the Service Registry. Service Provider will perform requests from a service consumer. Service provider interacts with a service code and executes the requests without knowing how consumer functions and does not even worry about it consuming services.

Service Registry is another component of SOA architecture that hosts service description. It is responsible for linking Services Requester and Service Provider.

Service Requester is an SOA Component which is responsible for discovering a service by searching through the service description given by the Service Registry and initiates a communication with Service Provider.

A standard Web Services Description Language (WSDL) is used for describing the web services information and publishing into Service Registry. The Simple Object Access Protocol (SOAP) is another standard that uses XML format for request/response between Service Requester and Service Provider. The three basic 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 to describe how exactly a specific web service works [5] based on the standards specified by the World Wide Web Consortium (W3C). The UDDI directory used as a registry of web services that is obtainable for use in a specified network. It would tell us where to locate the service and also to inspect the WSDL document. The SOAP specification provides standards for the format of a SOAP message and how SOAP should be used over HTTP. In the binding part of web services [6] SOAP is created to transport XML documents from one computer to another and while transport it can use a number of standard transport protocols.

The Representational State Transfer (REST) is an architecture design for the networked applications is what is being adopted these days. The REST Web Service approach uses REST only as a communication technology to build SOA. Requests and Response Services will be defined using SOA style decomposition and REST-based Web Services are used to transfer representations of resources as a transport. REST is implemented directly on top of the HTTP protocol because REST architecture is basically client-server architecture, and is designed to use a stateless communication protocol. Each resource accessed using Uniform Resource Identifiers (URIs) has one or more representation such as XML and JSON. JSON became popular due to advancement in AJAX technology on the client side. This way REST provide a better services to divergent platforms like android, java web applications, .Net Applications.

RESTful services are web applications built upon the REST architecture. RESTful services expose resources 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 analogies four main HTTP methods create, retrieve, update, and delete ( so-called CRUD actions). 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, provides message security by transport protocol (HTTPS) which offers only point-to-point security. 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.

The following are some of the proposed security implementation for RESTful web services

- 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

The main contributions and organization of this paper are summarized as follows: In section II we describe about literature review of clock skew optimization. The section III is about proposed work. Finally, in section IV the conclusions are mentioned.

## II. Related Work

Let's briefly understand the encryption techniques for algorithms – AES, Blowfish, RC2, RC4 and Rijndael

**AES:** The Advanced Encryption Standard (AES) is an encryption algorithm designed for securing sensitive data. Though it is unclassified by U.S. Government agencies, this eventually became an encryption standard for commercial transactions in the private sector. AES is designed based on a substitution-permutation or combination of both substitution and permutation, because of which it is fast in both software and hardware. AES does not use cryptographic technique called Feistel network like earlier DES algorithm. AES is one of the variant of Rijndael algorithm where it has 128 bits size fixed block and variants of key size of 128, 192 or 256 bits.

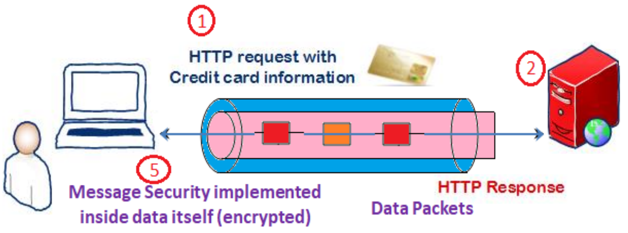
**Blowfish:** Blowfish is a symmetric key block cipher, which is designed to get included in a large number of cipher suites and encryption products. Blowfish provides an excellent encryption rate in software and no effective cryptanalysis has been found yet to date. Blowfish has a block size of 64-bit and a variable key length it takes up to 32 to 448 bits. It uses a 16-round Feistel cipher. Blowfish is a fast block cipher, even though it slows down when changing keys. Each new key requires pre-processing equivalent to encrypting about 4 kilobytes of text, which is very moderate when compared to other block ciphers. This intercepts its use in certain applications, but this is not a problem in others. Blowfish is known to be vulnerable to attacks on reflectively weak keys. One must select a key carefully because there is a class of keys known to be weak in Blowfish. At the same time using some keys may switch to more modern alternatives like the Advanced Encryption Standard.

**RC2:** In java encryption cryptography, RC2 is also known as ARC2. RC2 is also a symmetric-key block cipher and keys can be ranging between 1 to 128 bytes long.

**RC4:** RC4 is a stream cipher with symmetric keys. It was originally designed by Rivest for RSA Data Security (now RSA Security). This generates a pseudo random key stream that is used to generate cipher text by XOR with plain text. The sequence of bytes generated are not random, the output always be the same for given input but it is harder to crack because of approximated random properties.

**Rijndael:** The Rijndael encryption has become one of the Federal Information Processing Standard and is being accepted by developers. However, the designers have the algorithm of choice where encryption is required. This encryption algorithm has recently been published as the Advanced Encryption Standard (AES). Rijndael is a block cipher, rather than a streaming cipher, with data being processed in 128 bit blocks. Keys are longer than in previous systems, being 128 bits, 192 bits and 256 bits. The new standard of which Rijndael encryption is the basis, still in use as of 2010, is Advanced Encryption Standard (AES), sometimes called AES (Rijndael).

## III. PROPOSED FRAMEWORK



**Step 1)** first the Web Service generator who has the original WSDL file sends its encryption request to Secure_WSDL Web Service.

**Step 2)** the Secure_ WSDL Web Service gets the WSDL file and will do two things on the received original WSDL document (step 3 and 4).

**Step 3) Signing process:** Due to the fact that we need a strategy for the WSDL file receiver to make sure that it has received a copy of the original one. So in this step the Secure_WSDL Web Service will sign the WSDL document with its own private key to make authentication possible for the other side.

**Step 4) Encryption process:** Secure_ WSDL Web Service will encrypt the message and PortType elements of WSDL file. To fulfill the encrypting process, one can use either symmetric or asymmetric encryption. To do symmetric encryption the sender does not need to obtain recipient public key to do the encryption or in other words the sender does not need to know who the receiver of the WSDL file is. Because, sender has to obtain the receiver's public key by implying a type of key management infrastructure, so due to the fact that using asymmetric encryption is too expensive for both receiver and sender side the symmetric encryption seems to be more proper from many aspects. As it was said before if we use the symmetric encryption a secure channel will be required to transfer shared key between two sides.

When the WSDL file was encrypted it will be sent through internet to recipient. When the recipient received the encrypted file again the recipient will do two jobs on encrypted document (step 5, step 6) in order to obtain original WSDL file.

**Step 5) Decryption Process:** In this step first the receiver uses the respective key to decrypt the encrypted file. If the symmetric encryption has been applied on the encrypted file, the receiver will use his own private key to decrypt the document, or else if the symmetric encryption is used the receiver will use shared key to decrypt the encrypted document.

The performance of encryption was measured in terms of response time (Average execution for 1000 concurrent threads) and CPU is utilization. The result has been shown in graphical representation for Average execution time (Y-axis: milliseconds) and percentage of CPU utilization (Y-axis: percentage of CPU).

For XML size of 3KB input file with 1000 concurrent threads, it is observed that ARCFOUR encryption algorithm has been fastest and Blowfish seems to be the slowest in encrypting the XML. The CPU utilized for ARCFOUR which is the lowest and RC2 algorithm is the highest.

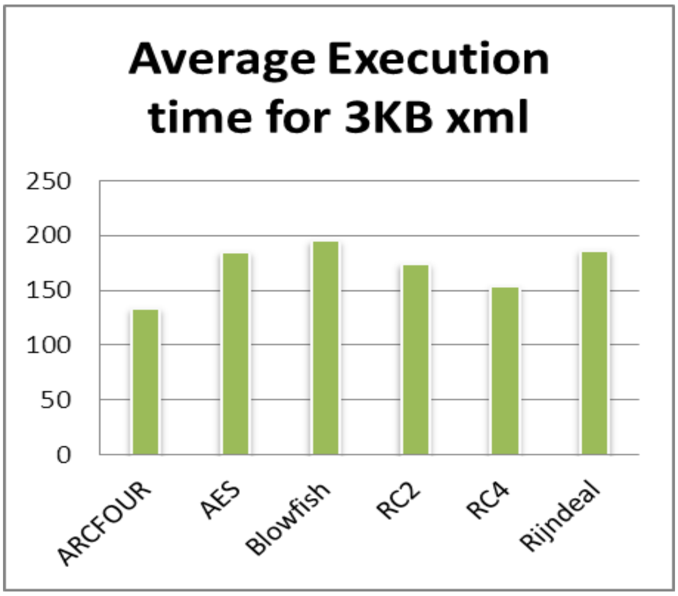


Fig 2: Average execution time of encryption schemes for 3Kb

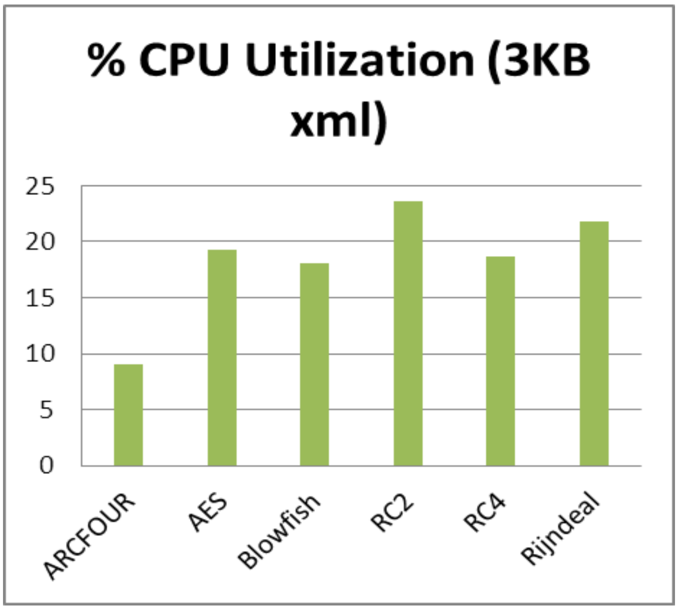


Fig 4: CPU utilization time of encryption schemes for 3Kb

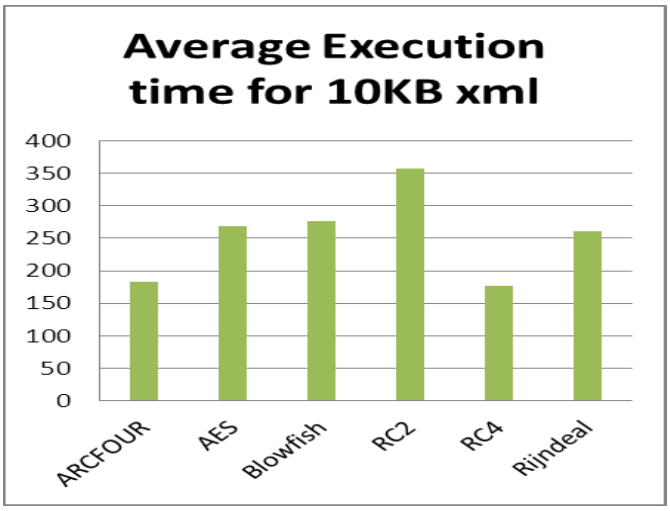
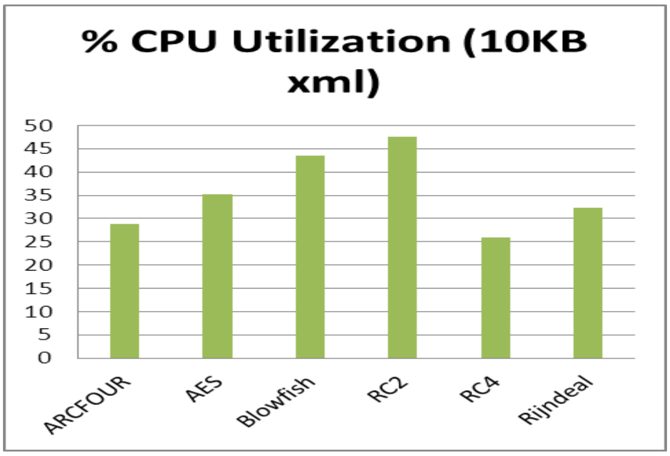


Fig 3: Average execution time of encryption schemes for 10Kb

For XML size of 10KB input file with 1000 concurrent threads, it is observed that RC4 encryption algorithm has been fastest and RC2 seems to be the slowest in encrypting the XML. The CPU utilized for RC4 is the lowest and RC2 algorithm is the highest.



## IV. CONCLUSION

The above case study showcase the performance of each of ARCFOUR, AES, Blowfish, RC2, RC4 and Rijndeal java crypto algorithms. The average execution time for encryption by each of the algorithm is increased with the data to be encrypted. However it is observed that RC4 encryption algorithm is very optimistic in resource utilization (in terms of CPU) and execution time. The encryption algorithm AES found to be expensive in overall performance

## References

[1] [1] Deven Shah and Dhiren Patel, "Architecture Framework Proposal for Dynamic and Ubiquitous Security in Global SOA," International Journal of Computer Science and Applications, Vol. 6, No. 1, pp. 40-52, 2009.

[2] Dirk Krafzig, Karl Banke, Dirk Slama, "Enterprise SOA Service Oriented Architecture Best Practices," Pearson Education, Inc, USA, 2005

[3] Johnneth Fonseca, Zair Abdelouahab, Denivaldo Lopes and Sofiane Labidi, "A Security Framework for SOA Applications in Mobile Environment," International Journal of Network Security & Its Applications (IJNSA), Vol.1, No.3, pp. 90-107, 2009

Fig 5: CPU utilization time of encryption schemes for 10Kb

[4] Hassan Reza, and Washington Helps, "Toward Security Analysis of Service Oriented Software Architecture," Proceedings of the 2011 International Conference on Software Engineering Research and Practice, Vol. II, 2011\

[5] [5] http://www.w3.org/TR/wsdl

[6] http://www.w3.org/TR/soap/