OPTIMIZATION OF SERVICE SELECTION AND MESSAGE SECURITY IN ENTERPRISE WEBSERVICE PLATFORM FOR INTEROPERABILITY

by

RAJNI MOHANA

a thesis submitted for fulfillment of the requirements

for the degree of

Doctor of Philosophy

in

Computer Science and Engineering

Under the Guidance of

Prof. DEEPAK DAHIYA

Department of Computer Science and Engineering
Jaypee University of Information Technology

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY
WAKNAGHAT, SOLAN-173234
INDIA

Roll No. 106201 **JAN, 2013**

CERTIFICATE

This is to certify that the thesis entitled, "**OPTIMIZATION OF SERVICE SELECTION AND MESSAGE SECURITY IN ENTERPRISE WEBSERVICE PLATFORM FOR INTEROPERABILITY**" which is being submitted by **RAJNI MOHANA** in fulfillment for the award of degree of Doctor of Philosophy in computer Science Engineering by the Jaypee University of Information Technology, is the record of candidate's own work carried out by her under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.

Dr. Deepak Dahiya

Professor

Computer Science and Engineering
Jaypee University of Information Technology
Waknaghat.

ACKNOWLEDGEMENT

"I bow in adoration unto the Great Spirit, the Principle of principles, the One who has no beginning, no middle, no end, the unborn, the One who knows no growth and no decay, the immortal."

Now, I have come to the end of my journey of Ph.D. and finally take this opportunity to express my gratitude to friends, colleagues, collaborators, relatives, etc. who have provided me the moral and scientific support for completion of this thesis.

First of all I would like to thank my supervisor, **Dr. Deepak Dahiya**. His actions have inspired me to dream more, learn more, do more and become more. His guidance, scientific skills, talent and knowledge has enabled me to attain the level where I find myself today. I am extremely grateful for his unfailing patience, readiness to help throughout by affectionate behavior, constructive criticism, useful advice and astute guidance.

I would like to express my sincere thanks to **Dr. Y Medury,** COO of Jaypee Education System. I am also thankful to our Vice Chancellor **Prof. Ravi Prakash** and Director**, Brig. Balbir Singh** for their invaluable support. Also I would like to offer my deep gratitude to **Prof. Satya Prakash Ghrera, HOD (CSE)** for approving my endeavor and imparting their beneficial suggestions. I am also thankful to **Prof. T.S. Lamba** for his ever challenging guidance to explore new ideas.

I am short of words in expressing my gratitude towards my husband Dr. Vimal Paul who have been and will be guiding star in my life. It is his inspirations and prayers which have always been a driving force behind all this. I am indebted to my father in law and mother in law for their unfailing patience, great tolerance, inspiration and lots of love and care. I know they will be the happiest persons to see my thesis in print. I am thankful to my Son Master Aryavir Paul and hope he forgives me for spending my time in the research which he deserved. Special thanks to my parents and brother for their guidance and support that I have been able to complete this arduous journey.

A word of thanks to the library staff of our university.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER I: INTRODUCTION

In the current scenario of research Service Oriented Architecture (SOA) can be described as one of the components of Enterprise Architecture (EA). EA came into existence from one of the most cited works by J. A. Zachman called as "Zachman Framework for Enterprise Architecture" developed in 1967**.** The EA provides a holistic view of an entire organisation. It is a coherent whole of principles, methods, and models that are used in the design and realisation of an enterprise's organisational structure, business processes, information systems, and infrastructure [1]. EA covers the overall construction of the enterprise in terms of business operations, finance, people, and buildings in addition to technology, and it covers technologies other than IT, such as for manufacturing or transport. The objective of EA is defined as

"The end object of Enterprise Architecture is not building and running systems, getting programs to run but is to engineer the enterprise so it is as LEAN as possible (minimum possible complexity and minimum possible costs) and MEAN as possible, that is, so that is can dynamically accommodate external demands." John Zachman [1]

The various elements in the EA and the relationship between various components are shown in figure 1.1.

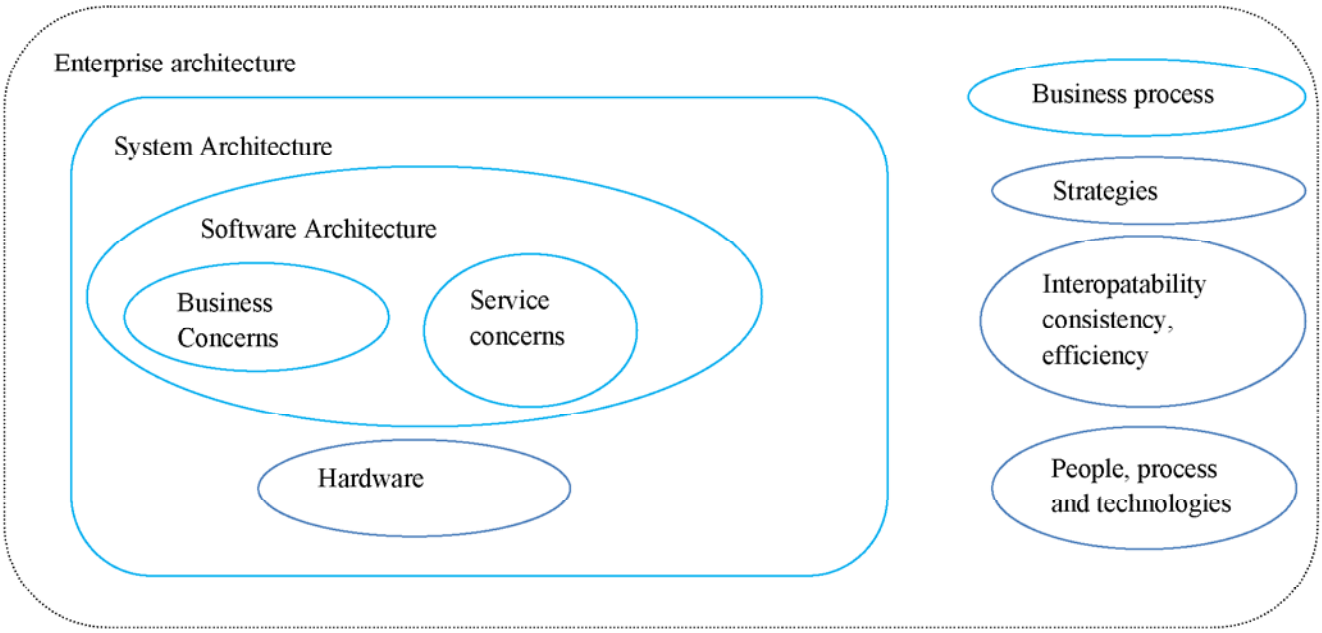


Figure 1.1: Enterprise Architecture framework for an enterprise

Enterprise architecture as shown in figure 1.1 consists of system architecture and strategies. It is the job of the architect to align system architecture with the strategies of the enterprise. The strategies can be

- Business process
- People, process and technologies
- Interopatability consistency & efficiency
- Political environment

System architecture details the way in which desired functionality is met by hardware and software components. It also figures out how the components relate to each other and the intended users of the system. As shown in figure 1.1, system architecture is composed of software architecture and the hardware.

Application and software architecture terms are used synonymously. Software architecture is the blueprint for the system as well as the project developing. It defines the work assignments that must be carried out by design and implementation teams. It assures that the application is built in right way and fulfills the objective of building the application. This normally includes things such as decomposing the application into its constituent classes and components, making sure design patterns are used in the right way, building or using frameworks, etc. It consists of both business and service concerns. The business concerns are defined as the functional or core concerns of the application while the service concerns are defined as nonfunctional concerns.

Software architecture to build an IT driven enterprise is shown in figure 1.2. In today's distributed computing environment, the code is distributed in the software application in the middle layer hereby termed middleware. The middleware ensures an n-layer architecture that adapts a "plug and deploy model" providing platform independence to various layers, flexibility in coding etc.

1.1 Service Oriented Architecture (SOA)

SOA [2] configures entities (services, registries, contracts, and proxies) to maximize loose coupling and reuse. It can also be defined as a deployment infrastructure on which new applications can quickly and easily be built. Organization for the Advancement of Structured Information Standards (OASIS) defines SOA as the following:

> *A paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. It provides a uniform means to offer, discover, interact with and use capabilities to produce desired effects consistent with measurable preconditions and expectations.*

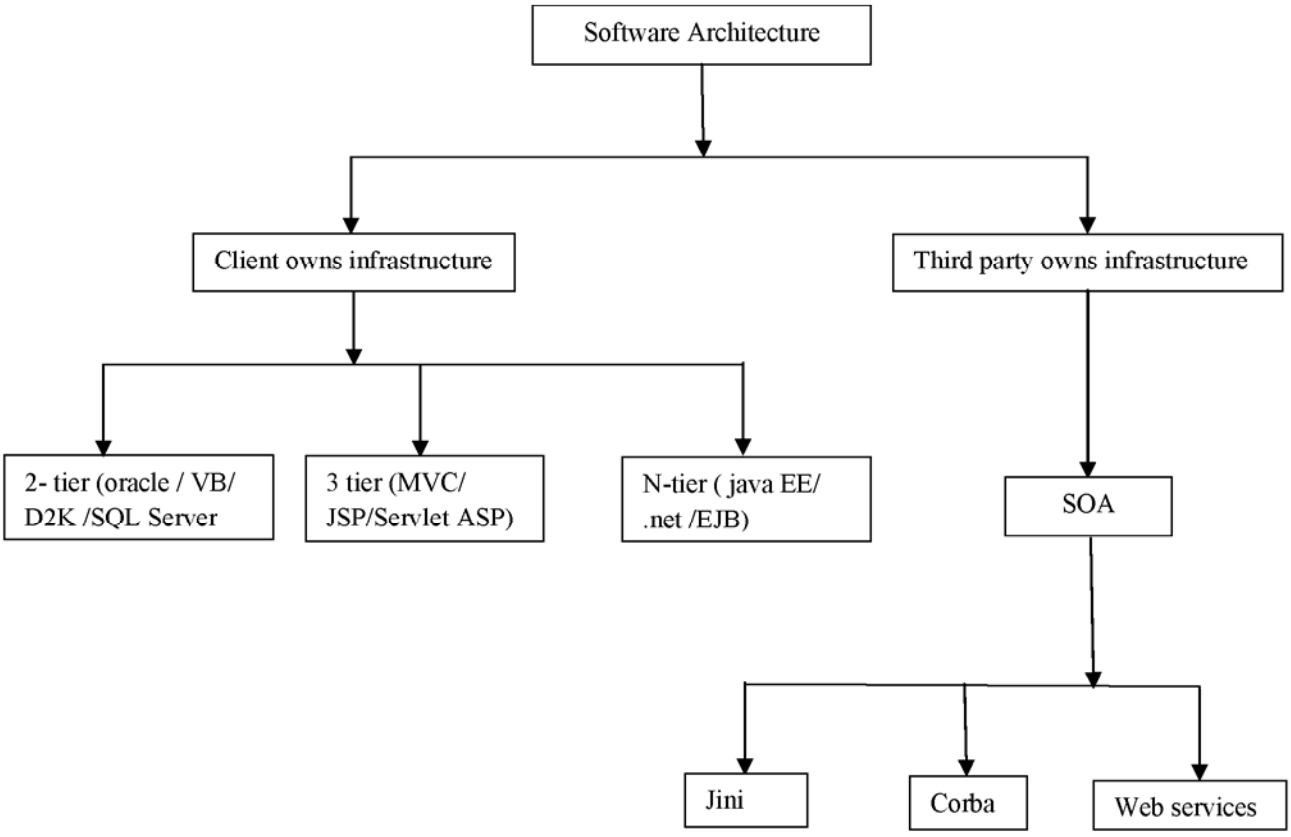


Figure 1.2: Architectures for IT Enterprise

J. Leon Zhao et al. [3] states in the paper **"Services computing as the foundation of enterprise agility**: **Overview of recent advances and introduction to the special issue"** that services computing is a **new research field since the year 2003** that goes beyond traditional computing disciplines as it includes not only architectural, programming, deployment, and other engineering issues, but also management issues such as business component modeling, business process design, and service delivery. **It explains the importance of service computing in the agile world.**

SOA as mentioned above can be implemented with web services. The web services [1] represent a type of relationships-based interactions (activities) between at least one service provider and one service consumer to achieve a certain business goal or solution objective [4].This relationship can be established by web service platform or enterprise service Bus. The important features of SOA are techniques like service composition, discovery, message-based communication, and model-driven implementation. These features are the reason for fast development of effective and flexible solutions in an enterprise.

In enterprise world for information dissemination, Service Oriented Architecture (SOA) plays an important role in developing Business to Business (B2B)/ Business to Consumer (B2C) applications with agility. Applications built on SOA requires interoperating software applications, running on heterogeneous platforms and frameworks like jini, Corba, Web services as shown figure 1.2. With SOA, the IT systems perform services that are defined and described in the context of the enterprise's business activities. This is the basis for the propagating the concept of system interoperability that SOA brings, not only within enterprises, but also between enterprises.

1.2 Achieving SOA

SOA are based on XML that is widely used for cross-platform data communication

- WSDL (Web Service Description Language) [5]: It used to describe the service and how a service requester can communicate with the web service. XML is used to define the public interface to the web service. These include protocol bindings and message formats required to interact with the web service listed in the directory.

- SOAP (Simple Object Access Protocol) [6]: It is based on XML that allows communication among the web service by sending a one way message. Message contains a structured data to be sent fro one process to another using any transport protocol like TCP / HTTP/ SMTP. The data is packed in an envelope and the communication is said to be completed when the recipient sends a reply back to sender.

- UDDI (Universal Description, Discovery, and Integration specification) [5]: The UDDI specifications define a way to publish and discover information about Web services. The service requester can search the data in the directory, which maintains WSAL files of all the services published by the service provider. UDDI creates a platform-independent, open framework for describing services, discovering business and integrating business services, using the internet.

Architecture for SOA requires providing connectivity among service providers and requesters, facilitating their interactions even if they are not exactly matched. The pattern can be implemented using a variety of middleware technologies and programming models. SOA can be designed using any of the two architectures given below:

1.2.1 Web Service Platform

Web services are a means for business to communicate with each other and with clients; it helps to provide business agility in a Business to Business (B2B)/ Business to Consumer (B2C) applications. It allows organizations to communicate data without intimate knowledge of each other's IT systems. Web services appear as a black box behind the firewall. Web services provide a standard means of interoperating between different software applications, running on a variety of platforms and/or frameworks. The Web Service Platform allows businesses to expose business assets as services. Standardizing interactions amongst services has the added advantage that any enterprise can out-source parts of its operation that it does not have expertise in. The implementation of a SOA is done using a web service platform which is shown in figure 1.3.

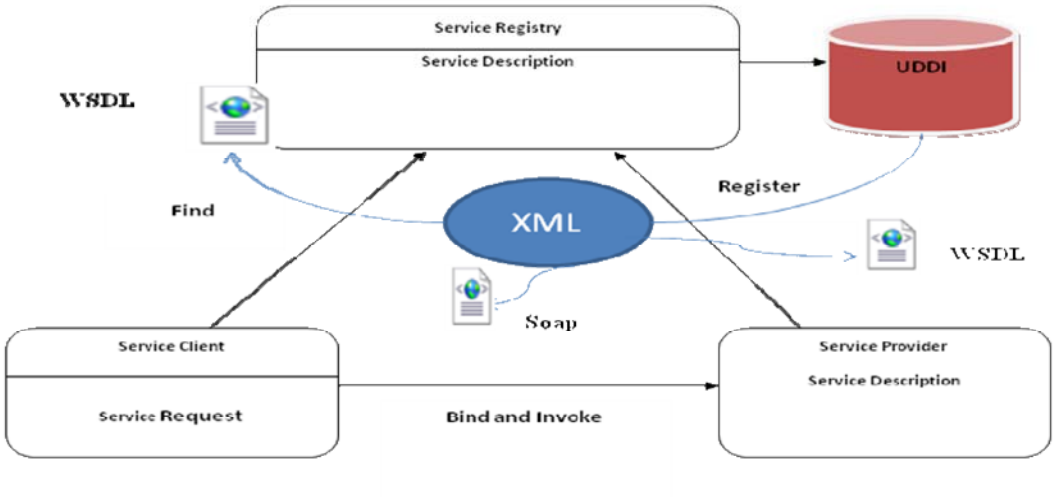


Figure 1.3: Web service Platform [7]

Participants in a Web services model are categorized into three types

- Service Requester : Finding a web service of interest is an important task for a service requester
- Service Provider: Publishing their web services in registries is the task of service provider
- Service Registry: It maintains the details of all the web services which are published by the service requester. It also enables the service to lookup for the best service according to its requirements among large number of functionally-equivalent services.

The web service platform allows service requesters and service providers to interact in consistent manner independent of the underlying software domains and enforce business rules and policies such as data validation rules, service level security, service-level management, service level agreements . It also allows an SOA to scale up an enterprise –wide business requirements. To provide interoperatability the basic web services platform consists of specifications (SOAP, WSDL and UDDI). In order to generate composite services, it is required to discover and select suitable web services for service requests. Some standards like Web Service Definition Language (WSDL) can be used to describe functional aspects of a Web service in form of a service description which is advertised in some Universal Description, Discovery and Integration (UDDI) registries [7].

1.2.2 Enterprise Service Bus

Application built on SOA requires interoperating software applications, running on heterogeneous platforms and/or frameworks, which can be achieved by using Enterprise Service Bus (ESB) Platform. ESB is a middleware which provides a means for business to communicate with each other and with clients. It is an infrastructure for SOA service connection and message connection without intimate knowledge of each other's IT systems [6]. Figure 1.4 shows SOA in ESB platform. It consists of service requesters, service providers, ESB gateway, ESB name space Directory, Business service directory.

Business Service Directory is design-time directories or customized service directories which give the details of the various services published by the service providers in the Zone [6]. When a service requester looks for a service in BSD a service discovery protocol is used to find out the best web service among various functionally equivalent services.

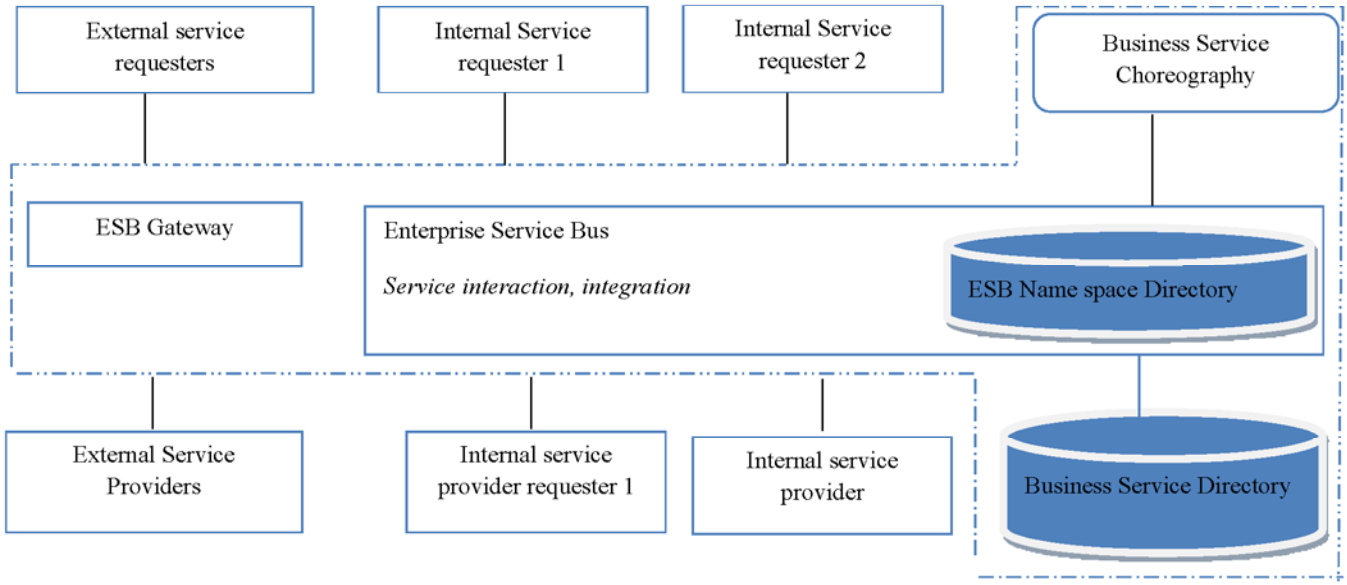


Figure 1.4: Enterprise Service Bus Platform in SOA [3]

1.3 Applications of SOA

SOA is applicable in many applications like healthcare, e-governance or any Business to Business (B2B) or Business to Consumer (B2C) applications. Let us consider a scenario of a B2C application like online shopping (figure1.5), seeing how it's implemented using a web

services. When a client puts a request on the portal, a SOAP message is sent to the inventory to check for the stock, if it's available then the details of it is appended to the SOAP message, and the message is further forwarded to billing department. The billing department further prepares the invoice according to the incoming SOAP message, appends the bill information to the SOAP message and sends the message to the shipping department, which takes care of shipping the details to the right person by placing the order to the manufacturer's web service.

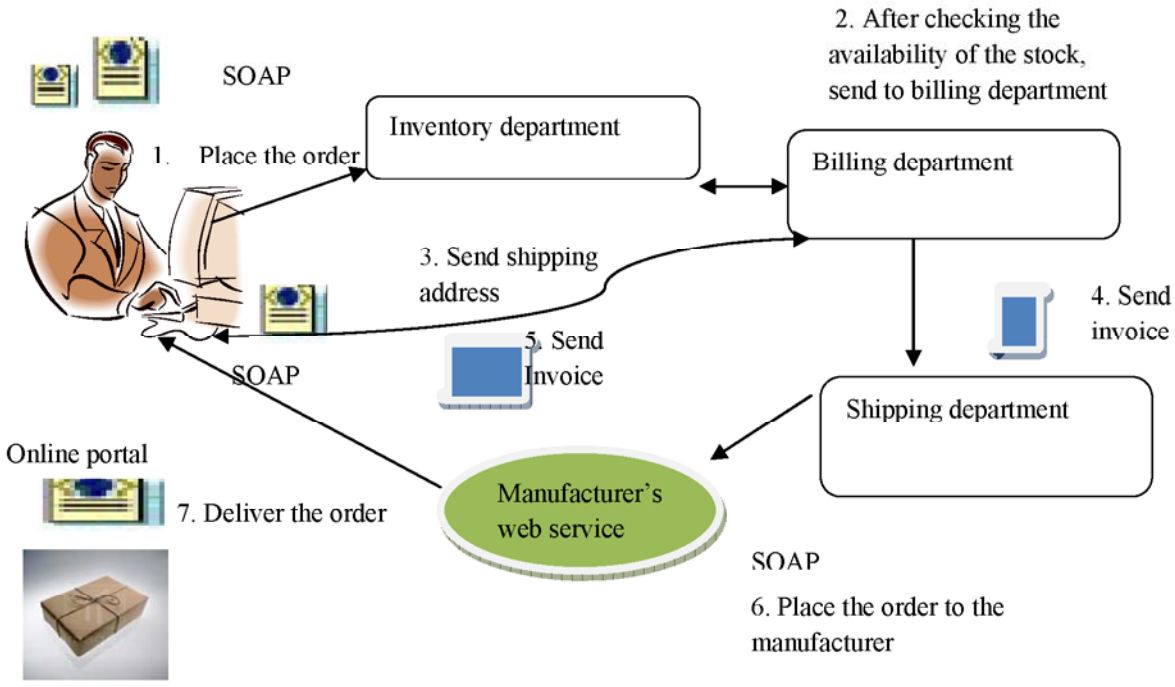


Figure 1.5: Online shopping Scenario

1.4 Research Domains in SOA

Adopting a SOA needs developing a service strategy that takes into account its business drivers, context, and application domain (the problem space). The implementation needs various decisions to be made in various domains as shown in figure 1.6 namely business, engineering, operations and cross cutting domains [8]. Cross cutting domain has affect in all the domains as it deals with all the non-functional concerns of the applications. The cross-cutting domain includes such areas as training, education, and governance. Our problem space is the engineering domain.

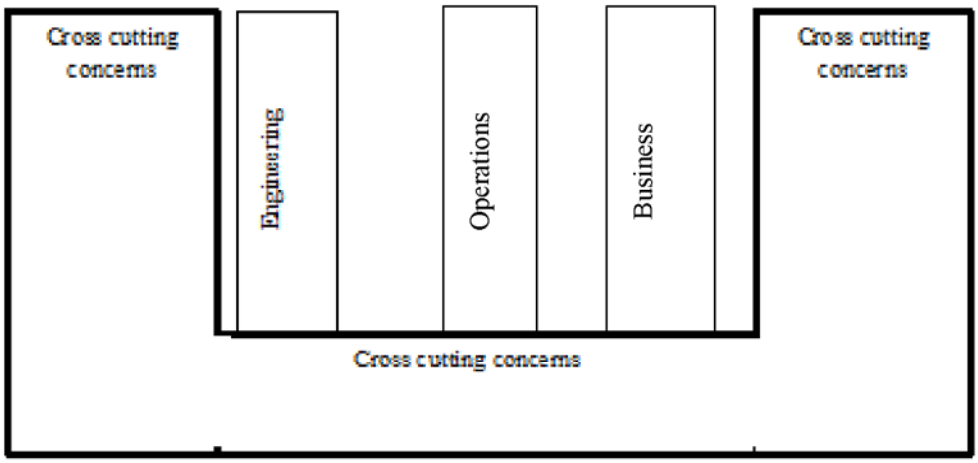


Figure 1.6: Research Domains in SOA

The business domain deals with the various forms of service orientation and its impact in the organization. Some of the drivers for service-oriented systems in this domain are the needs to be on-demand, customizable, trusted, compliant, agile, and measurable. In the context of SOA the engineering domain deals with the service-oriented system life cycle [9] as one of the main aspects. The service-oriented system life cycle comprises of various phases like process models that can be used to build service-oriented systems, requirement models for denoting functional and non-functional aspects, platform- and computational-independent architectural abstractions, design patterns, logging, model-driven code generation, verification, testing, and maintenance.

The operations domain deals with the various operations, its evaluation and optimization of service-oriented systems. Some of the drivers in this domain are for service-oriented systems to be ambient, user friendly, high impact, pervasive, and adoptable. This domain includes aspects like adoption, monitoring, and support of deployed service-oriented systems. Cross-cutting concerns are issues like governance, training, education which affects all the other domains.

1.5 Advantages of SOA

The integration of SOA applications can reduce the dependency of different types of IT systems, reduce the cost of system maintenance and the complexity of the IT system operation, increase the flexibility of the system deployment, and at the same time exclude the barrier of service innovation [10].

It is the replacement of large, monolithic applications that have tiny interoperability interfaces, grudgingly provided and not guaranteed, by smaller, modular services that have interface descriptions and contracts. This is one of the fundamental contributions of SOA.

Reusability is the most frequently quoted reasons for SOA adoption as in today's real world it's important to provide business agility in a B2B / B2C applications. For business agility we need to capture and reuse the experience of the IT architects in such a way that future engagements can be made simpler and faster. We do this by capturing knowledge gained from each engagement and using it to build a repository of assets. IT architects can then build future solutions that are based on these proven assets. This reuse saves time, money, and effort, and helps ensure delivery of a solid, properly architected solution.

The features which make it a first choice to implement SOA are [3]:

- Pervasiveness
- Simplicity
- Platform-neutrality
- Scalability
- Business agility

1.6 Disadvantages of SOA

SOA Architecture would not be suitable for applications with GUI functionalities. Application requiring heavy data exchange would become more complex if implemented using SOA. An application which requires asynchronous communication can't make use of SOA [6]. It also becomes an added burden for standalone and short lived applications. So a wise choice is to be made before selecting the architecture for any software application.

1.7 Organization of the Thesis

The thesis is organized as follows:

Chapter I of the thesis has introduces the concepts of SOA and how it fits in the overall structure of enterprise architecture. It presents an overview of research domains of SOA. The chapter also lists out the advantages and disadvantages of SOA.

Chapter II discusses the related study of the research work. It outlines the concepts of service selection, XML processing and security. It outlines how the new paradigm has emerged to play a significant role in service selection, XML processing and security as a result of the growing demands of today's software practitioners and applications. Furthermore, it provides the motivation for this line of research along with the main research challenges of this study. At the end the research gaps are listed and based on that objective of the proposed research work are formulated.

Chapter III presents the optimized business service directory. The proposed business service directory is a fuzzy expert system. The rules for the expert system are automatically generated using particle swarm optimization. The dataset used to train the swarms is QWS dataset. The chapter also contains the details of the implementation of the model. At the end results and observations are discussed.

Chapter IV discusses about rewriting attack and how it leads to insecure conversation. A SOAP Model is proposed with three recommendations. It also discusses the implementation of the model and present how early detection of rewriting attacks are done, thus ensuring secure conversation. At the end the overhead of the recommendations was discussed and a comparison with respect to various existing models.

Chapter V discusses the proposed secure dissemination technique. It discusses how XML encryption can be done using DNA encryption. The technique is computationally secure where as the overhead of this technique is very less. The technique is implemented as a publish/ subscribe multicast interface.

Chapter VI concludes the thesis by drawing together the main arguments of this work and summarizing the contributions that have been made. It also outlines the future scope of work that could be carried out based on this line of research.

CHAPTER II: RELATED STUDY

As discussed in the previous chapter the research domain in SOA spans over engineering domain, business domain, operation domain, crosscutting concerns domain. The research work generates the problem specification from the engineering domain. In the context of SOA the engineering domain deals with the service-oriented system life cycle as one of the main aspects. The service-oriented system life cycle comprises of various phases like process models that can be used to build service-oriented systems, requirement models for denoting functional and non-functional aspects, platform- and computational-independent architectural abstractions, design patterns, logging, model-driven code generation, verification, testing, and maintenance.

The research work addresses some of the important issues of the late requirements and early design stages. This chapter primarily focuses on completed and ongoing work in service selection, XML processing and security.

2.1 Service Selection

When a service request is issued, the requirement specifications are looked for in the UDDI / ESB as shown in figure 1.1 to find services that can provide expected functionality. It is the job of service discovery protocols to minimize the administrative overhead and increase usability by locating the best web service among the large number of functionally equivalent web services, published in UDDI. Research work on various centralized discovery approach as proposed by researchers [11-18] was analyzed during the literature survey.

The research work in the area of service selection protocols is directed in two different areas namely, semantic and non-semantic approaches both of which are focused around discovering the web service based on QoS (Quality of Service) in brief.

- Semantic approach: It performs web service searching based on the domain knowledge of the attributes. It requires prior systematic structuring of the contents such that requests are understood and responded based on their meaning.
- Non-semantic approach: It performs web service searching based on the syntax. It requires the data to be structured as per required format and the requests will be responded accordingly.

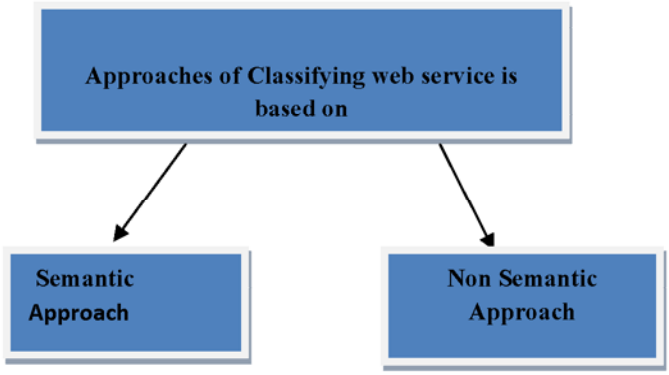


Figure 2.1: Classification of approaches of service discovery

2.1.1 Semantic Approach

Segev, A. et al. [11] analyzed different methods for automatically identifying possible service composition. They explored two sources for service analysis which are commonly used in service repositories i.e.

- WSDL description files
- Free textual descriptors,

They also investigated two methods for Web service classification for each type of descriptor:

- Term Frequency/ Inverse Document Frequency (TF/IDF)
- Context based analysis, and a baseline method.

Their work showed that the web-based context extraction method by analyzing both the WSDL description and the textual description yields better results than the TF/IDF method and string matching.

In addition, the results prove the advantage of integrating the analysis of both the WSDL context descriptor and the service textual descriptor [11].

Kritikos. k et al. [12] have proposed QoS-based *WS Description and Discovery (WSDD)* to enable automatic discovery and composition of independently developed and deployed (web) services. They have considered semantically rich QOS based in automatic discovery of Web Services (WSs) can be achieved by incorporating semantics into WS matchmaking and selection (i.e., discovery) process. Its main contribution is the analysis of the requirements for a semantically rich QoS-based *WSDM* and an accurate, effective QoS-based *WS Discovery*

(WSDI) process. They also provided a road map to extend *WS* technologies* for realizing semantic, functional, and QoS-based WSDl.

This approach [12] comprises of

a) Description of the QoS aspect of WSs (i.e., QoS-based WS description)
b) Filtering of WS functional results based on user constraints on their QoS descriptions (i.e., QoS-based WS matchmaking) The algorithms were implemented using Mixed Integer Programming MIP, CP, and explanation-based CP
c) Sorting the results based on user-provided weights on QoS attributes/metrics (i.e., WS selection).

Pilioura et al. [13] have proposed *PYRAMID-S* which uses a hybrid peer-to-peer topology to organize web service registries based on domains. It has a scalable framework for unified publication and discovery of semantically enhanced services over heterogeneous registries. In such a topology, each registry retains its autonomy, meaning that it can use the publication and discovery mechanisms as well as the ontology of its choice.

The features of *PYRAMID-S* are:

— unified Web service publication and discovery:
— over heterogeneous registries, thus alleviating the users from the burden of handling the diversion between different technologies;
— based on syntactic, semantic, and Quality of Service (QoS) information, improving in this way the precision and the recall;
— preservation of the autonomy of Web service registries by allowing the accommodation of different publication and discovery mechanisms; and
— Use of a scalable infrastructure which organizes registries based on domains

Yin Baocai et al. [14] proposed an approach to qualify a web service using its semantic description. They suggested a general ontology to describe the non functional qualifies the web service using the semantic description of the web services using their non- functional specifications. This ontology can solve the interoperability of QoS description. The framework supports the automatic discovery of web services and it can improve the efficiency for users to find the best services. It uses *OWL-S* as the description language to represent QoS ontology.

2.1.2 Non-Semantic Approach

Zhang et al. [15] has designed a broker-based architecture called *QBroker*, to provide end-to-end QoS management for distributed services. Functionalities of QBroker include service discovery, planning, selection, and adaptation. The efficiency of QBroker is dominated by the running time of the service selection algorithm. They also designed efficient algorithms for quality-driven Web service compositions. Their model defines multiple QoS criteria and takes global constraints into account. It ensures that the selected services always meet the QoS requirements. They have also proposed heuristic algorithms to find near-optimal solutions in polynomial time which is more suitable for making runtime decisions. They have mapped the service selection to a 0-1 multidimensional multi choice knapsack problem (MMKP) [15].

Vuong Xuan Tran et al.[16] have suggested an approach where the web services are selected based on QoS. The web services are ranked based on their quality attributes, but the issue raised by the researchers was that it's difficult to provide a precise value to the quality attributes of the web service. Thus they suggested using fuzzy logic to support using imprecise QoS constraints. The benefit of this approach is that a user does not need to mention crisp values of QoS properties. Instead the user can use fuzzy linguistic concepts to express their expectation of service quality.

However a user has to define at most as many rules as there are degrees of acceptance that one wants to differentiate. When a number of QoS properties are involved, the number of rules can be large and it becomes a tedious task for the user. It does not consider that some QoS criteria can be defined by using only crisp form such as criteria having Boolean or string value type [16].

Maolin Tang et al. [17] quoted service discovery as so-called optimal web service selection problem. This paper proposes a new hybrid genetic algorithm for the optimal web service between some web service implementations like dependency and conflict constraints. When an implementation is selected for one web service, a particular implementation for another web service must be selected. This is so called *dependency constraint*. Sometimes when an implementation for one web service is selected, a set of implementations for another web service must be excluded in the web service composition. This is so called *conflict constraint*. Thus, the optimal web service selection is a typical constrained combinatorial optimization problem from the computational point of view. The hybrid Genetic Algorithm (GA) has been

implemented and evaluated. They compared various techniques used for service discovery like penalty-based genetic algorithm, the repairing-based genetic algorithm and the hybrid genetic algorithm. It is shown that hybrid genetic algorithm is better than above mentioned techniques. The hybrid genetic algorithm is more suitable for those web service problems with a large number of abstract web services and a large number of constraints [13].

Table 2.1: List of advantages and disadvantages of various soft computing techniques [19]

Soft computing technique used in non semantic approach	Advantages	Disadvantages
Artificial Neural networks[18]	• High accuracy for float values	• Easy to over fit • Rules hard to extract and hard to understand
Genetic Algorithms [17]	• It updates the population and search for the optimum with random techniques.	• It is difficult to implement because of crossover and mutation. • System doesn't guarantee success
Fuzzy Logic [16]	• User does not need to specify concrete values of properties.	• A user has to define at most as many rules as there are degrees of acceptance that s/he wants to differentiate

Al-Masri et al. [18] has used Artificial Neural Networks (ANN) to classify the best web service according to the user requirements. The ANN classifies the web service based on QoS. They have designed a Web Service Crawler Engine (WSCE) that provides an active monitoring tool that continuously collects the most recent and up-to-date QoS values. They have also provided a QWS data set generated using WSCE for the purpose of research. They obtained most Web services from public sources on the Web, including UDDI registries, search engines, and service portals. The dataset consists of 5,000 Web services, each with a set of nine QWS (Quality of Web Services) attributes that we measured using commercial benchmark tools [18].

2.1.3 Analysis of the Service Selection Protocols

While analyzing non semantic approaches it was observed that it involves soft computing techniques like *Fuzzy Logic, GA, ANN* to discover the best web service based on QoS. Their advantages and disadvantages are discussed in Table 2.1.

Subsequent research work [16] using fuzzy approach states that the problem while using only fuzzy is that a user has to define at most as many rules as there are degrees of acceptance that s/he wants to differentiate. When a number of QoS properties are involved, the number of rules can be large and it becomes a tedious task for the user. Additionally, for each QoS property, there are number of membership functions being modelled but they may not satisfy some users.

2.2 Rewriting Attacks and Secure Conversation

Another challenge which our research work addresses **XML processing and security.** The communication among these web services is established using SOAP messages which are based on XML. XML is the most relevant means to provide interoperatablity among various entities. When in network a XML file can be prone to hacking and unauthorized access, thus affecting the data integrity and confidentiality which are supposed to be important issues of communication. Data integrity and confidentiality can be breached by rewriting attack and insecure dissemination.

The rewriting attacks leads to insecure conversation as the contents of a SOAP message protected by an XML Signature as specified in WS-Security can be altered without invalidating the signature [20].

The work in [21] provides a detailed analysis on the possible scenarios that enable these signature rewriting attacks. They figured out the limitations of the existing solutions and proposed a solution that uses a subset of XML Path Language (XPath) called Fast*XPath* instead of *ID* attributes for signature referencing in web services messages, and have shown that this solution is both efficient and secure. Their approach recommended considering the absolute path from a signed element to the document's root element ("vertical fixing") and to its siblings ("horizontal fixing"). The position of the signed element must be fixed so that it is not feasible to move the signed element without invalidating the reference—and thus the signature. As it requires to explicitly naming every single element on the path from the document root to the signed subtree, there is no flexibility

available to move any signed contents to another location within the document.

The disadvantage highlighted by them is that *FastXPath* poses severe restrictions to the abilities of defining a signature reference, it turns out to be a trade-off between security and flexibility, as every flexibility within the reference can potentially be exploited for a wrapping attack [21].

In [22] the authors have listed out limitations of existing solution and accordingly have recommended three solutions as part of their work to solve the problem of XML rewriting attacks. The recommendations are listed below:

1. The first recommendation was to take into account the depth information of signed object and send this information in the header along with the SOAP message.
2. Their second recommendation keeps information regarding the parent of a signed element, if an element is copied and pasted in the bogus header then the information regarding its parent name will not be the same and the attack will be detected
3. Finally, the third recommendation is the usage of this *Id* attribute to uniquely identify a parent of an element. *WS-Security* specification also specifies that two instances of *Id* within a document cannot have the same value.

The disadvantage of this work is that the depth information can be tampered and *Id* referencing is not safe as it can also be tampered by the malicious attacker.

The work in [23] has suggested an approach to be applied for early detection of XML rewriting attacks particularly targeting the secure SOAP-based conversations. This work suggested that at the time of sending SOAP messages either in a trust scenario and/or in a secure conversation scenario, one can always keep an account of structure of the SOAP elements by including the following information into the SOAP account header (not exhaustive):

- Number of child elements of the root (Envelope).
- Number of header elements.
- Number of references for signing element.
- Predecessor, Successor, and sibling relationship of the signed object.

The above solution can detect XML rewriting attacks but it may not comply with the schema of the WS* standards and might even violate further processing of XML messages.

[24] showed that a SOAP account suggested by [23] is itself vulnerable to XML rewriting attacks. They suggested that a module can be created which can keep track of the fact that the received SOAP message has a SOAP account header. If it is there then the module will verify the signature of the SOAP account. If several intermediaries have their own SOAP account then there will be a nested signature as it is described. If verification is successful then the module will do the rest of the routine work.

The above suggestions increases the level of verification with nested signature, hence the process will be slow as signature generation is a slow process.

Table 2.2: A quick summary of work done by various researchers

Researchers	**Work description**	**Tradeoff**
[21] Sebastian Gajek, Meiko Jensen, Lijun Liao, and J¨orgSchwenk	Recommended Fast*XPath*	Fast*XPath* is not flexible as it limits the abilities of defining a signature reference [21]
[22]AzzedineBenameur	Recommended using depth information, parent information and using *Id* attribute to uniquely identifying the parent.	Sending these information over the network is also not same, as it can also be tampered by malicious attacker [16]
[23] Mohammad AshiqurRahaman, Andreas Schaad	Recommended maintaining the information and sending it with the SOAP message • Number of child elements of the root • (Envelope). • Number of header elements. • Number of references for signing element. • Predecessor, Successor, and sibling	It may not comply with the schema of the W S * standards and might even violate further processing of XML messages [24].
[24] Yong Liu, Haixia Zhao, Yaowei Li	Suggests signing the SOAP account , if the intermediaries have their own SOAP Account then there will be a nested signature	The process will be slow as signature generation is a slow process.
[25] Smriti kumar Sinha, AzzedineBenameur	Suggest using Context-Sensitive Signature (CSS)	The trade off is that the context is to be generated and stored in the reference element of the signature in header section before signing the message[5].

Another work [25] proposes a formal solution to XML rewriting attack on SOAP messages using Regular Tree Grammar (RTG). The authors in [25] have used Context-Sensitive Signature (CSS), which captures the context of the signed element at the time of signing and incorporates the signed message context in the signature element making any change detected. The authors provided algorithms to generate the context and to verify it.

The trade off is that the context is to be generated and stored in the reference element of the signature in header section before signing the message. Table 2.2 provides a quick summary on the work of the various researchers.

In [26, 27] the work highlights various threats on SOAP Messages like replay attack, parameter tampering, rewriting attack etc., and have proposed an Integrated Application and Protocol Framework (IAPF) that can successfully combat the security threats. IAPF helps in the early detection of both XML injection and parameter tampering attacks.

2.3 Secure Dissemination

Secure dissemination [28] of an XML file is one of the techniques to ensure data integrity and confidentiality. The research work presents a secure dissemination technique which ensures that extraneous is inaccessible even if the consumer is a legitimate consumer. Consequently, this avoids information leak. Recent years has seen remarkable progress in recent years to address access control policies for secure dissemination of XML file.

Earlier researchers [29, 30], have implemented secure dissemination using structural based routing. The routing model presented by them is based on multi-casting of document portions from an intermediate router to the subscribers. Essentially the router may send the some document portion multiple times to the subscriber. Another disadvantage is if the Local XML structure changes, solutions given by [29, 30] requires associated routing topology to be changed. Thus a subscriber needs to have a prior knowledge of the routing structure as the router structure as the router cannot fetch any content which is not hosted currently.

In [5] a centralised publish/ subscribe middleware is presented which is able to perform selective XML content delivery based on a shared ontology. It suggests semantic queries over domains concepts to compute a set of candidate concepts and over publisher's policies to check evolving policies for a subscriber. In [6-10] presents of the work as a request response paradigm in client- server architecture.

2.4 Research Gaps

A few limitations of the existing research work in this area are:

i. When a number of QoS properties are involved, the number of rules can be large and it becomes a tedious task for the user, to specify the rules.

ii. Existing solutions of rewriting attacks either don't comply with Ws Standards, or are not efficient in detecting the rewriting attacks

iii. The existing solution of the secure dissemination doesn't support an n-ary tree. Reconstruction of the tree in the consumer end is also difficult.

2.5 Motivation

The motivation for this research work is the outcome of the study of the relevant literature that pinpoints to the research gaps in addressing some of the important issues of the late requirements and early design stages i.e., **Service Selection** and **XML processing and Security.** The rules to locate the best web service should be automatically generated by using a dataset. One more aspect that ought to be considered is the reduction in the number of rules by eliminating the rules having zero weight age according to the dataset. Another motivation is to ensure data integrity and confidentiality by successfully detecting the rewriting attack and ensuring secure dissemination.

The overall research work is defined in terms of the following three problem statements

a. To propose a Business Service Directory for the ESB Platform for efficient and fast service selection.

b. To propose SOAP Model for efficient detection of rewriting attacks and ensuring secure conversation

c. To propose a Secure Dissemination Technique that keeps a check on information leakage.

2.6 Objectives and Scope

The research work attempts to bridge the gap in the SOA development life cycle primarily focusing on the issue of service selection and security. The three proposed artifacts namely business service directory (BSD), SOAP model and secure dissemination technique fulfills the stated objectives of better service selection and security these objectives are summarized as follows:

i. To ensure better service selection among the various functionally equivalent services.
ii. To reduce the time required to search the web service
iii. To design a system where the rule to search the service should be adaptive in nature.
iv. To remove rewriting attacks such that if an authorized person has modified the content pretending someone else as the constructor, then it can be detected.
v. To ensure secure conversation such that end-to-end integrity can be established
vi. To encrypt the soap message securely
vii. To ensure the secure dissemination of the XML data such that only those consumers who are authorized to read the data can read it.

2.7 Summary

This chapter introduced selected work and developments across the broad spectrum of service selection, XML processing and security research. It provides a comprehensive overview of state of the art in the field and exhibits a number of open problems that drive continuous research into SOA. At the core of the chapter is the bifurcation of research work into service selection, XML processing and security.

The following chapter III, IV and V discusses the above issues separately as three component of our research work. The next chapter discusses the proposed optimized business service directory that addresses the issue of service selection.

CHAPTER III: OPTIMIZED BUSINESS SERVICE DIRECTORY FOR THE ESB PLATFORM

This chapter discusses the optimized business service directory for the ESB platform. Enterprise service bus is a middleware which provides dependable and scalable infrastructure that connects heterogeneous applications and mediates their interactions, and makes them broadly available as services for additional uses. As shown in figure 1.4 Business Service Directory (BSD) is maintained to provide the details of the various services publisher in the zone. To establish a connection between the service provider and service consumer of a business application, Service search and selection is to be performed in BSD. Selecting the best web service among the various functionally equivalent web services, published in BSD is a complex problem.

The major challenge is regarding the changes that arise in QoS of the web services. Thus the optimized BSD is designed as a fuzzy expert system which can adapt dynamically to respond to such changes.

3.1 Background

The following section presents a brief introduction of fuzzy logic, fuzzy expert system, fuzzy clustering and optimization technique PSO. These concepts are used later in the chapter to implement optimized service discovery protocol in service registry.

3.1.1 Fuzzy Logic

Applying fuzzy logic is beneficial for defining QoS description and measuring quality parameters in order to compute the overall QoS. Inference methods are used when the input-output relation can be expressed in the form of if-then rules. We take advantage of the fuzzy logic for measuring overall QoS.

Aristotle [37] gave the theory of Boolean logic, which is two valued logic: true and false. Lofti Zadeh extended it to handle the concept of partial truth by presenting Fuzzy Logic, where the truth value may range between completely true and completely false [38]. It states that the variable values can be represented by degrees representing its closeness to truth. These types of variables are called as *linguistic variables* for example temperature, humidity etc. The values of the linguistic variables called as *linguistic values* are represented in the form of degrees for example temperature can be high, low or medium. These degrees may be managed by specific functions called as *membership functions. The membership function* is a

graphical representation of the magnitude of participation of each input. It associates a weight with each of the inputs that are processed, define functional overlap between inputs, and ultimately determines an output response.

There are many types of membership function [38] but we will be using Z, S and Triangular.

The rules in the fuzzy rule base use the input membership values as weighting factors to determinetheir influence on the fuzzy output sets of the final output conclusion. Once the functions areinferred, scaled, and combined, they are defuzzified by the defuzzification interface into a crisp output which drives the system as shown in figure 3.1. This system is called as *fuzzy expert system [39].*

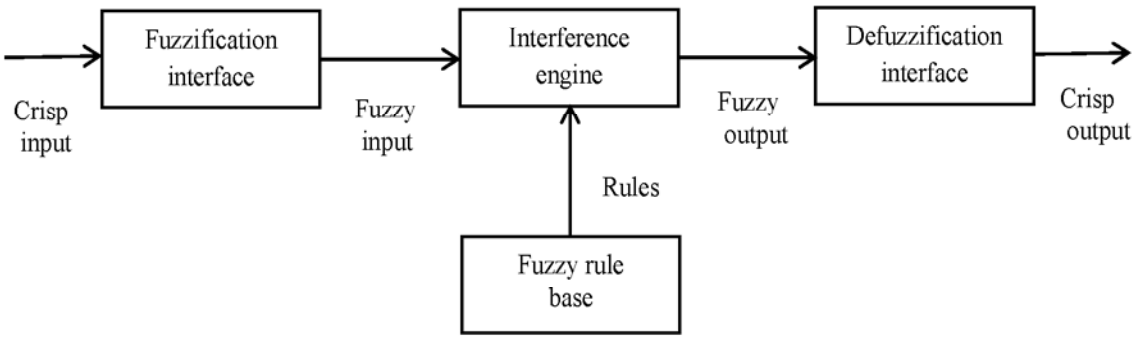


Figure 3.1: Fuzzy expert system

A fuzzy expert system [39] is application software that performs a task that would be performed by a human expert. It simply uses a collection of fuzzy membership functions and rules, instead of Boolean logic, to reason about data.

3.1.2 Fuzzy C- means Clustering

Clustering is a mathematical tool that attempts to discover structures or certain patterns in a dataset, where the objects inside each cluster show a certain degree of similarity. In fuzzy clustering, data elements can belong to more than one cluster, and associated with each element is a set of membership levels. These indicate the strength of the association between that data element and a particular cluster. Fuzzy c-means (FCM) [40] clustering is a process of assigning these membership levels, and then using them to assign data elements to one or more clusters. This technique was originally introduced by Jim Bezdek in 1981. FCM is a data clustering technique where each data point belongs to a cluster to some degree that is specified by a membership grade as an improvement on earlier clustering methods. It provides a method that shows how to group data points that populate some multidimensional space into a specific number of different clusters.

Areas of application of fuzzy cluster analysis include for example data analysis, pattern recognition, and image segmentation. The detection of special geometrical shapes like circles and ellipses can be achieved by so-called shell clustering algorithms. Fuzzy clustering belongs to the group of soft computing techniques (which include neural nets, fuzzy systems, and genetic algorithms) [41].

3.1.3 Particle Swarm Optimization

QoS parameters are often changing due to dynamic and volatile service environment. In such environment, Web Services need to be able to adapt dynamically trying to respect the service interaction. Changes are required to be captured; evaluated and proper actions need to be taken accordingly. Particle Swarm Optimization (PSO) [42] is used to optimize the rules according to the training data as explained in the algorithm explained in section 3.4.

Our research work proposes PSO, because of its advantages over other soft computing techniques.

The advantages of PSO are:

- Easy to represent the interaction between attributes
- Consider several attributes once
- Balance between local exploitation and global employment
- PSO is easy to implement and there are few parameters to adjust.
- All the particles tend to converge to the best solution quickly even in the local version in most cases.

PSO came into origin from the research of food hunting behaviours of birds. Researchers found that in the course of flight flocks of birds would always suddenly change direction, scatter and gather [43]. It is used to solve a wide array of different optimization problems. Their behaviours are unpredictable but always consistent as a whole, with individuals keeping the most suitable distance. Through the research of the behaviours of similar biological communities, it is found that there exists a social information sharing mechanism in biological communities.

Each swarm of PSO can be considered as a point in the solution space. If the scale of swarm is N, then the position of the i-th (i=0,1, 2. ..N) Particle is expressed as Xi. The "best" position passed by the particle is expressed as pBest [i]. The speed is expressed with Vi.

The index of the position of the "best" particle of the swarm is expressed with g. Therefore, swarm i will update its own speed and position according to the following equations [42].

$$Vi=w* Vi +c1 *rand ( ) *(pBest[i] *Xi)+c2*Rand)*(pBest[g] -Xi) \quad (1)$$

$$Xi =Xi+ Vi \quad (2)$$

Where c1 and c2 are two positive constants, rand ( ) and Rand ( ) are two random numbers within the range [0, 1], and w is the inertia weight, i refers to the swarm.

In other words, proposed BSD will lead to the design of a fuzzy expert system which will automatically perform service lookup. Fuzzy expert system consists of four components as shown in figure 3.1. The four components are fuzzification interface, inference engine, defuzzification interface and rule base. So optimized service registry component should also have all these fourcomponents to behave as a fuzzy expert system

The significance of our approach is that it solves the above mentioned issues by generating rules automatically by using a dataset; it also reduces the number of rules by removing the rules having zero weight age according to the dataset.

3.2 Proposed Business Service Directory for the ESB Platform

Architecture of proposed Business Service Directory for the ESB platform is shown in figure 3.2. The proposed Business Service Directory is a fuzzy expert system. The fuzzification interface converts crisp input into fuzzy input, which is passed through inference engine. The inference engine ranks the published web service based on the rules in the Rule base. The rule base consists of automatically generated rules (using training data) by fuzzy clustering and PSO. The defuzzification interface defuzzifies the fuzzy output into crisp output. The rules make the system automatic. These rules can be chosen by the human expert but our approach generates them automatically from the training dataset, thus it requires no human intervention in writing the rules for the system. The service environment is dynamic and volatile, which leads to changes in their respective QoS [2] parameters. In such an environment, web services needs to be able to adapt dynamically trying to respect the service interaction.

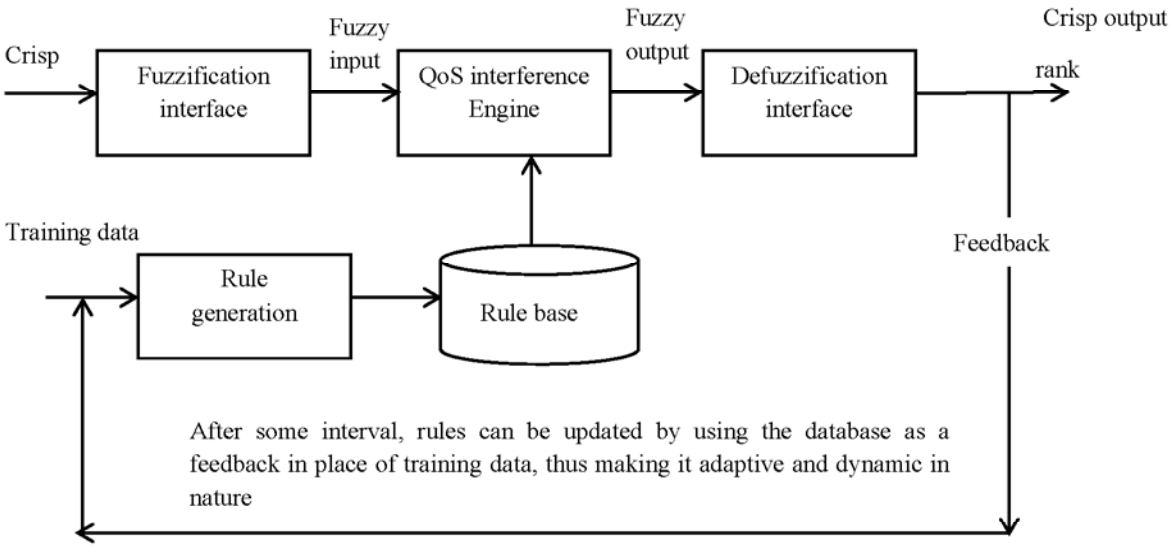


Figure 3.2: Architecture of the proposed Business Service Directory for the ESB Platform

The Changes in QoS of the webservices are required to be captured and evaluated. Later proper actions are taken accordingly. Thus the rules can be updated by using the database as a feedback in place of training data, which makes it adaptive and dynamic in nature. The Proposed Business Service Directory is presented by means of UML class diagram in figure 3.3.

The working of the proposed Business service directory can be very well explained using the information flow.

3.3 Information Flow for the Proposed Business Service Directory

The information flow for the proposed business service directory is described below:

1. If the user is a service provider who wants to publish his service in the BSD will require doing following steps:
   a) The service publisher will fill the form of input parameters which are quality attributes of the web services and submit it to business service directory
   b) In the business service directory, the input are accepted and ranked according to the rules by following a process of inference and defuzzification.

c) The input details of service publishing form and service rank is then stored in the database i.e., business service directory in a WSDL format

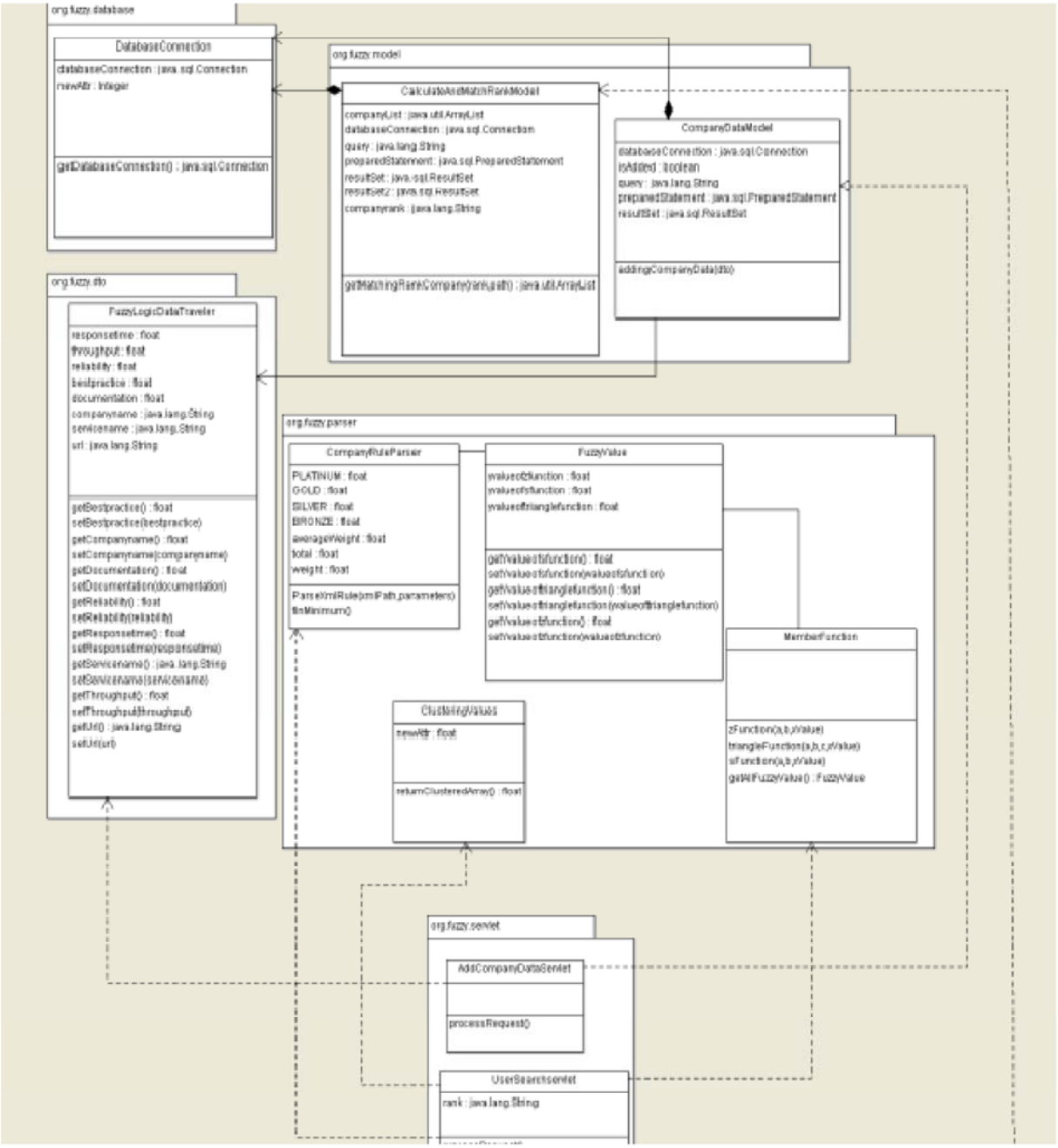


Figure 3.3: UML diagram of proposed Business service registry

2. If the user is a service requester

a. The service requester will fill the form of input parameters which are quality attributes of the web services and submit it to business service directory

b. In the business service directory, the input are accepted and ranked according to the rules by following a process of inference and defuzzification in decision making engine.

c. The business service directory is then searched for the web service that provides requested service and has the requested ranking.

d. The details of all the matching web service which were stored in the database by the service publisher is provided in the database

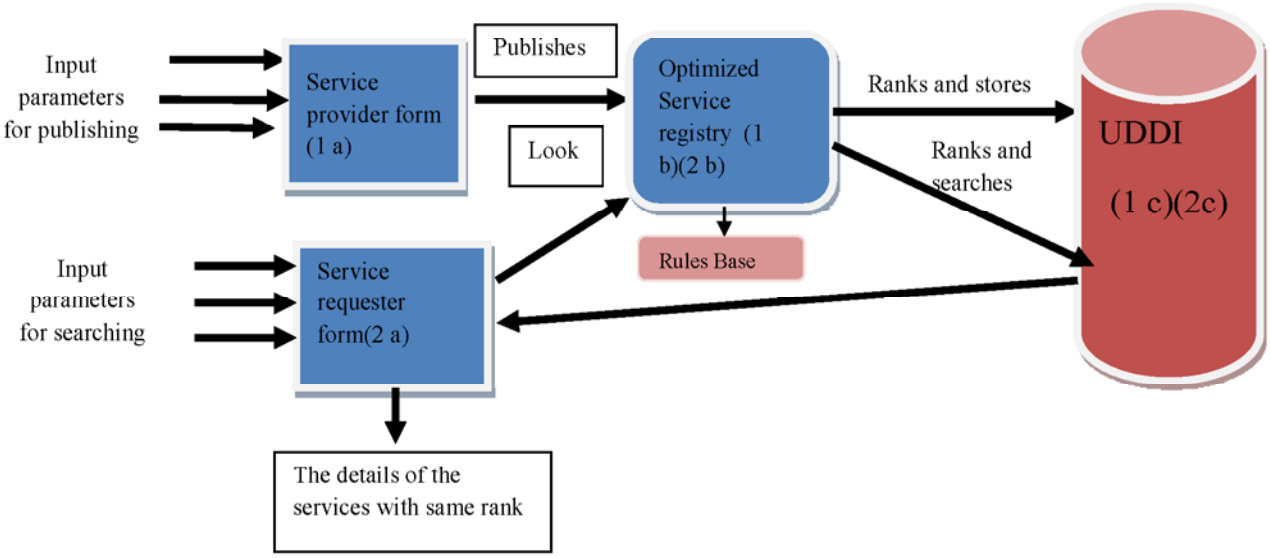


Figure 3.4: Information flow for optimized service registry component

3.4 Detailed Design for the Business Service Directory

The Designing of Business Service Directory presented in figure 3.4, encompasses two steps.

a) Generating optimized rules using the training dataset for performing service discovery in the Proposed Business Service Directory.

b) Designing the inference engine using a java EE platform, which triggers the rules to rank and match the request with the rank of the service published by the service provider.

3.4.1 Automatic Generation of Rules from Dataset

To rank the web services, the rules can be provided by the expert in the domain or alternatively it can be generated using the algorithm given below. The rules are generated using a dataset. The dataset used is called as training set. The technique given below also

generates lesser number of rules i.e., it removes all the rules which are having zero impact on the system.

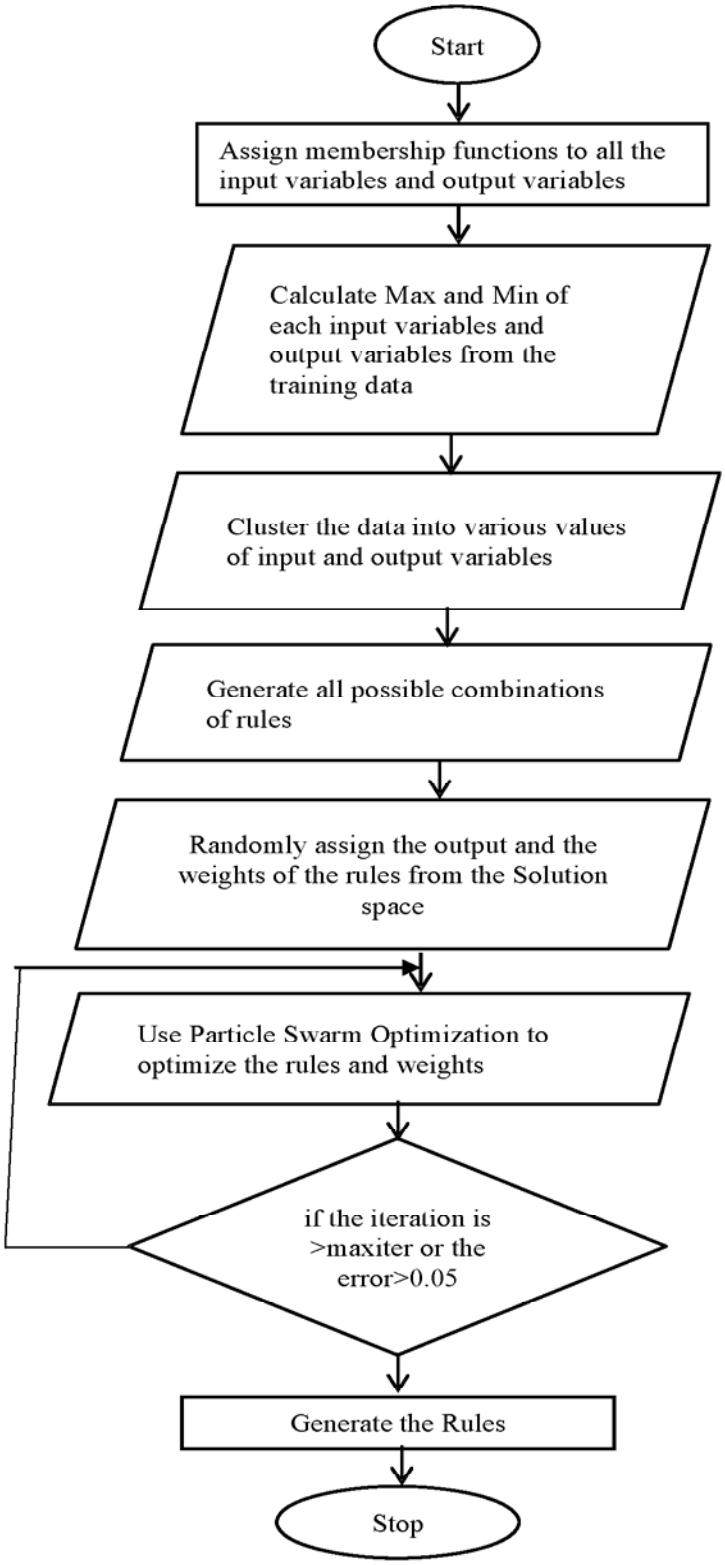


Figure 3.5: Flow chart for fuzzy rule based model

The Flow chart for fuzzy rule based model is shown in figure 3.5 and its corresponding algorithm is described below:

The above problem domain which involves calculating the rank of a web service based on the values of QoS parameters can be considered as TSK (Takagi, Sugeno & Kang) model. The shape of membership functionsfor the input variables are Z, S and Triangular respectively. The output is constant with parameters ranging from 0 ->1. The notations used in the algorithm are described as under

- Population is the no of swarms.
- Pbest is the best value obtained in each iteration.
- Gbest is the best value obtained among all the pbest.

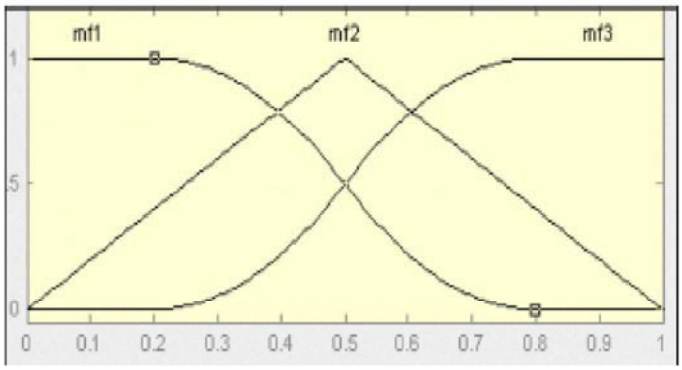


Figure 3.6: Membership functions for input QoS Parameters.

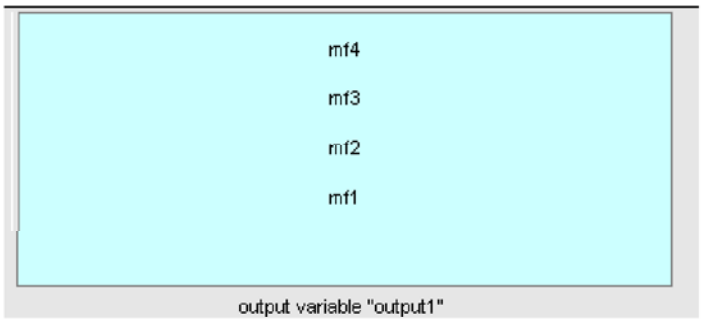


Figure 3.7: Membership functions for output QoS Parameters.

The algorithm for fuzzy rule based model is

i. Assign membership function to all the input variables .If there are two linguistic values of thevariables then assign S to the lower on and Z to the higher one. If the no of values are more than twoassign triangular membership functions to all the vales lying between higher and lower. Theparameters for S= [0.2, 0.8]

Z= [0.2, 0.8]

Triangular = [0, 0.5, 1]

As shown in figure 3.6

ii. Assign membership values to the output variables If values <=2 assign parameters 0 to low and 1to high respectively. If values > 2 assign parameters 0.5 to all the intermediate values as shown in figure 3.7.

iii. Calculate the max and min of each input parameter and output parameter from the data set used totrain the particles.

iv. Using FCM technique of fuzzy clustering assign the center points to the to membership function ofinput and output variables

v. Create a fuzzy inference engine containing all the possible combination of rules formed from inputvariables

Initialize the variables used in equation 1 and 2

i.e., Population size= 10

Maximum iteration = 100

wmax = 0.9

wmin = 0.3

c1 =2 ,c2 = 2

vi. Assign randomly an output to the rule set and optimize the result and calculate the deviation from the result called error.

vii. Pbest=result

viii. Pbestfitness= error

ix. Gbest= Minerror

x. velocity V is also assigned a random value

xi.. Vmax = 1

xii. Repeat steps from 12 – 18 till iteration <maxiteration& flag == 0(flag checks whether the output is in acceptable limit.

*VI=w*VI+cl *randO*(pBest[i]-Xi) +c2*RandO *(pBest[g] –Xi*
Xi =Xi + Vi
X is considered as the result of the rule set which is optimized by the particles.

xiii. Calculate the output according to the input in the training set.

xiv. Match the output; calculate the deviation from the result

xv. pbest=X
Pbestfitness= error
Gbest=pbest
GbestFitness= min of gbest

xvi. if the deviations is in acceptable limit mark flag=1

xvii. Assign the value of gbest as the consequent of the rules.

xviii. Weight /degree of each rule is equal to the maximum value of membership value of the input variables

Table 3.1: Web service quality attributes [44]

Parameter Name	Description	Units
Response Time	Time taken to send a request and receive a response	ms
Availability	Number of successful invocations/total invocations	%
Throughput	Total Number of invocations for a given period of time	invokes/second
Successability	Number of response / number of request messages	%
Reliability	Ratio of the number of error messages to total messages	%
Compliance	The extent to which a WSDL document follows WSDL specification	%
Best Practices	The extent to which a Webservice follows WS-I Basic Profile	%
Latency	Time taken for the server to process a given request	ms
Documentation	Measure of documentation (i.e. description tags) in WSDL	%
WsRF	Webservice Relevancy Function: a rank for Webservice Quality	%
Service Classification	Levels representing service offering qualities (1 through 4)	Classifer
Service Name	Name of the Webservice	none

In order to train the particles publicly available QWS dataset [45, 46] is used. The QWS dataset consists of information of QoS of various web services. The QoS which are listed in the dataset are shown in table 3.1. The main goal of this dataset is to offer a basis for Web service researchers. This dataset is collected considering a subset of 365 real web service implementations that exist on the Web today. The services were collected using our Web service Crawler Engine (WSCE) [45]. The majority of Web services were obtained from public sources on the Web including Universal Description, Discovery, and Integration (UDDI) registries, search engines, and service portals. The public dataset consists of 365 Web services each with a set of nine QWS attributes that we have been measured using commercial benchmark tools.

Each service was tested over a ten-minute period for three consecutive days by the researchers [44]. The various quality attributes considered are shown in table 3.2:

Table 3.2: Data set on searching the web service phone [44]

Name	Response Time (ms)	Throughput (hits/sec)	Reliability (%)	Best Practices (%)	Documentation (%)	Rank
DOTSGeoPhone	126.2	12.3	78.7	80	86	Platinum
Phone	150.45	7.4	82.1	82	37	Gold
DOTSPhoneAppend	118.5	0.7	70.2	80	90	Gold
PhoneVerify	131	1.6	65.9	72	41	Gold
PhoneNotify	437.62	1	68.4	69	93	Silver
PhoneService	133	1.4	64.7	82	10	Bronze
Phonebook	464	3.1	43.2	80	2	Bronze

Out of the above mentioned quality attributes a data set of five quality attributes based on the availability in QWS dataset was considered as input variables.

In the research work, the dataset for *web service phone* was considered

I. Response Time (ms) = {high, average, low}

II. Throughput (hits/sec) = {high, average, low}

III. Reliability (%) = {high, average, low}

IV. Best Practices (%) = {high, average, low}

V. Documentation = {high, average, low}

One parameter is considered as output variable

I. Rank ={ Platinum (High quality) ,Gold ,Silver ,Bronze (Low quality)}

The range for linguistic values (high, average and low) of QoS is calculated by performing fuzzyclustering on the training data. A data set which was generated using the demo [44] and searching for a *web service phone* is shown in table 3.2.

3.4.2 Designing the Inference Engine

The inference engine uses the above generated rules to rank the web services published by the service provider. It will then save the details of the web service in a database. If a request arrives from a service requester, it will be also ranked by the inference engine and based on the criteria of the service name and rank, the request will be searched. If a match found the details as well as the web address of the web service is provided to the service requester. The service requester can then directly communicate with the web service and establish the connection.

The registry is implemented using tools and technologies comprising of NetBeans IDE, JSP, Java EE, MySQL, Apache Tomcat, and XML. Servlet is used to process user request and JSPs is used to create the view which are the core component of the project. This project follows Model View Controller (MVC) architecture which keeps the code clean and manageable. Through MVC the processing, view and database code are kept separate.

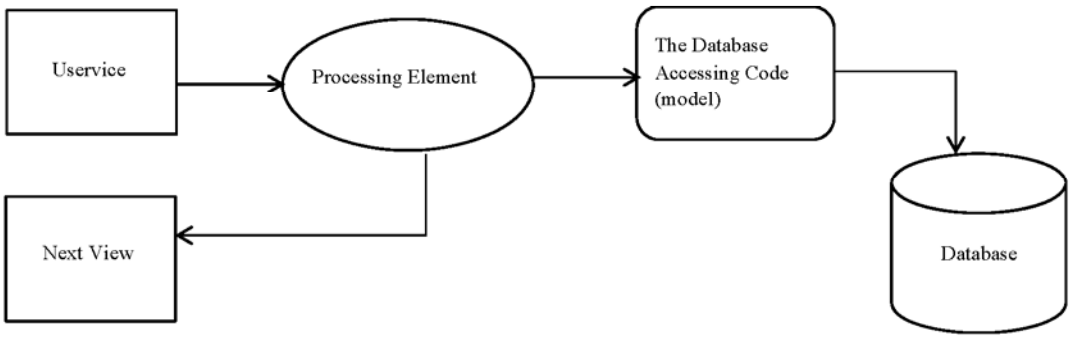


Figure 3.8: Architecture of the business service directory

The User view will constitute of a fuzzy rule parser. All the rules are stored in an XML file. Fuzzy rule parser triggers the rules when a request or a publishing takes place. XML is chosen for database accessing code because it requires less time than database hit and it's easy to manipulate. XML has an advantage that a large number of data can be stored and can be manipulated easily.

Defuzzification is done using weighted average technique. The registry provides a publishing API. The web service publishers can publish their web services on the basis of some parameter. Initially all the given parameters are stored in a database so that there should not be any delay in processing. Later all the data are converted into XML, as searching in the database consumes more time than searching in the tree structure of XML. Database chosen is MySQL. The enhanced entity relationship (EER) diagram in figure 3.9 describes the relationship and attributes in the various tables of database.

3.5. Implementation of the Optimized Business Service Directory

The fuzzy expert business service directory is implemented in two phases:

a. Generating the rules by implementing the algorithm described in section 3.5.1

b. Implementing the inference engine described in section 3.5.2

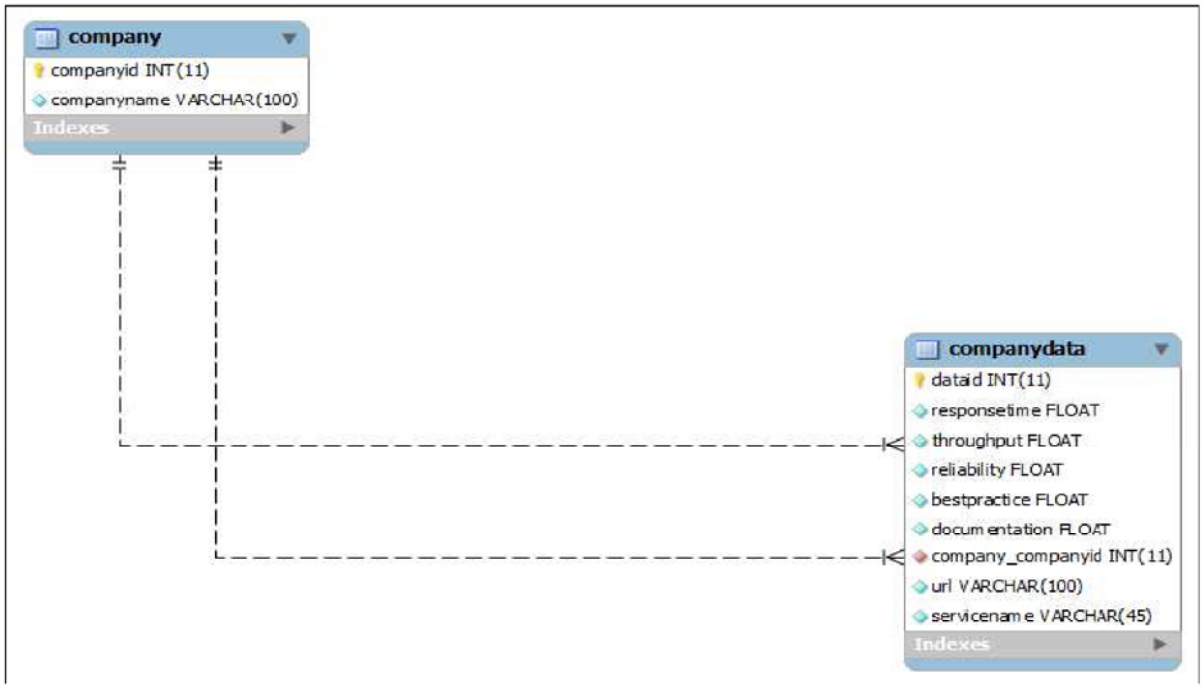


Figure 3.9: EER diagram of used database

First the rules are generated using the QWS dataset and then an inference engine is developed for the optimized business service directory which has a service requester and provider form. The data when fed on the form will process according to the rules using the designed fuzzy rule parser.

The output of the rule is then defuzzified to give a crisp output. The crisp output is the rank of the web service. If the rank is calculated for the service provider, the application will store all the details of the web service along with the rank.

When the service requester is looking for a web service in the database, his requirement is also ranked by the above mentioned technique, and then the web service with the specified rank is located in the database. If the search is successful, then the details of the web service stored in the database are provided to the service requester.

3.5.1 Automatic Generation of Rules

Rules generated after implementing the algorithm explained in section 3.4.1 using Matlab 7.0 is shown in table 3.3. The number of rules generated is 26. The number of rules generated (26) using the rule based model are less in comparison to generated by using Fuzzy Logic i.e., 243. Lesser the number of rules to trigger lesser is the seek time required to locate the best web service. The rules given above are stored in the rule base.

Whenever a service request comes from a service requester, it places his requirements, for example if a phone service which is highly reliable and has average response time is requested. The inference engine will rank the request according to the rules given in the rule base and match it with the rank of services registered by their service providers. Once a match is found the service requester will be given the network address of the selected service provider. The web service search lists all the details of the services and service provider. As there can be various services that gives similar service with same ranking. It now depends on the service requester to pick one among them and perform service composition.

Table 3.3: Rule generated from the data set and there corresponding weight

1. if Response Time is low and throughput is low and reliability is average and best practices is low and documentation if low then output is bronze 0.121455
2. if Response Time is low and throughput is low and reliability is average and best practices is low and documentation if average then output is silver 0.467807
3. if Response Time is low and throughput is low and reliability is average and best practices is average and documentation if low then output is silver 0.034978
4. if Response Time is low and throughput is low and reliability is average and best practices is average and documentation if average then output is silver 0.134724
5. if Response Time is low and throughput is low and reliability is average and best practices is average and documentation if high then output is platinum 0.000811
6. if Response Time is low and throughput is low and reliability is average and best practices is high and documentation if low then output is platinum 0.847967
7. if Response Time is low and throughput is low and reliability is average and best practices is high and documentation if high then output is gold 0.439231
8. ……
9. ……..
10. ……..
23. if Response Time is high and throughput is low and reliability is average and best practices is low and documentation if high then output is platinum 0.573544
24. if Response Time is high and throughput is low and reliability is high and best practices is low and documentation if high then output is silver 0.426456
25. if Response Time is high and throughput is average and reliability is low and best practices is average and documentation if low then output is platinum 0.000285
26. if Response Time is high and throughput is average and reliability is low and best practices is high and documentation if low then output is platinum 0.154625

3.5.2 Implementation of Inference Engine

The designed Business Service Directory consists of the inference engine using a Java EE platform. The controller shown in figure 3.9 consists of a fuzzy rule Parser. As we know Directory provides web service on the basis of rank. The rank is calculated on the basis of some parameters e.g. response time, throughput etc., supplied by the web service publishers. The rules are as under in an XML format:

```
<!ELEMENT response-time EMPTY>
<!ELEMENT throughput EMPTY>
<!ELEMENT reliability EMPTY>
<!ELEMENTbestpractice EMPTY>
<!ELEMENT documentation EMPTY>
<!ELEMENT rank EMPTY>
<!ATTLIST response-time value (high | low | average) #REQUIRED>
<!ATTLIST throughput value (high | low | average) #REQUIRED>
<!ATTLIST reliability value (high | low | average) #REQUIRED>
<!ATTLISTbestpractice value (high | low | average) #REQUIRED>
<!ATTLIST documentation value (high | low | average) #REQUIRED>
<!ATTLIST rank value ( platinum | gold | silver | bronze) #REQUIRED>
<!ELEMENTrule(responsetime,throughput,reliability,bestpractice,documentation,rank)>
<!ELEMENT rules (rule+)>
```

To parse the XML a DOM parser is used. DOM parser provides accessing of XML element in a hierarchical way. This makes accessing easier and faster. All XML elements are loaded simultaneously in memory and can be accessed in any order.

3.5.3 Working of the Proposed Business Service Directory

The tools used for implementing the component comprised of NetBeans IDE, JSP, Java EE, MySQL, Apache Tomcat and XML. This section presents the snapshots of the implemented Components.

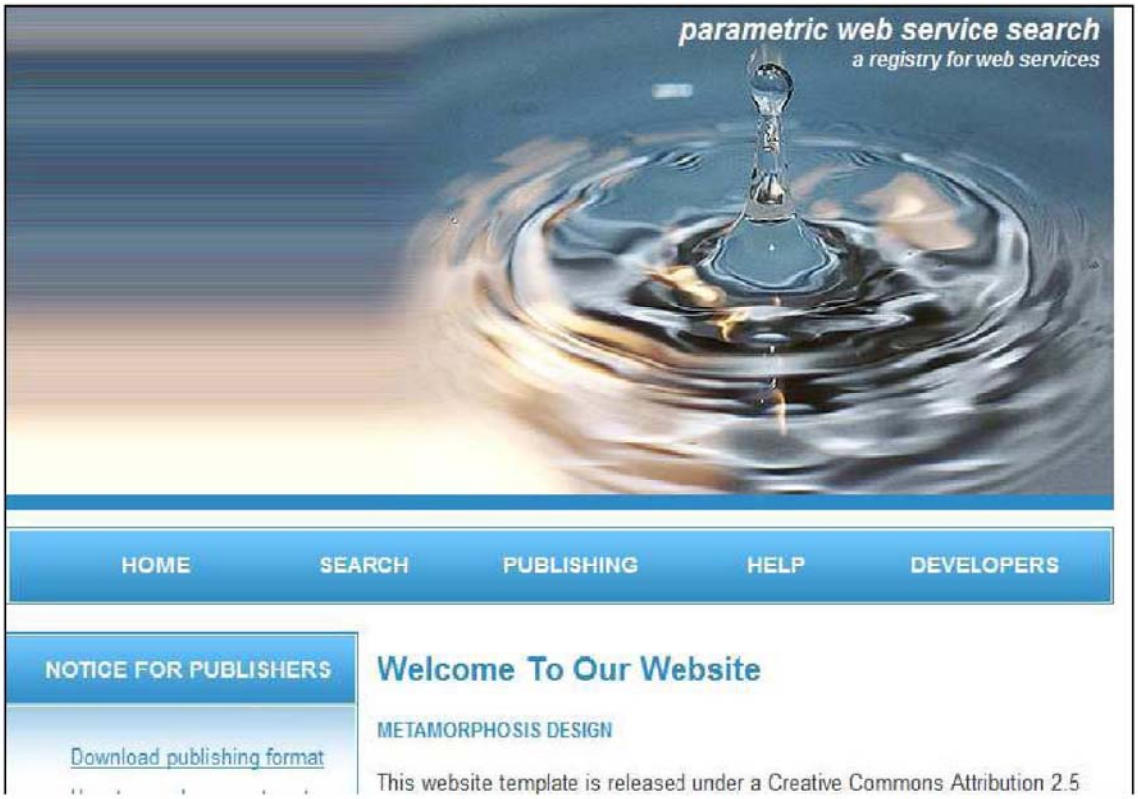


Figure 3.10: Snap shot of the home page

When clicked on publishing icon by the service provider the following form appears. The provider will have to fill the details of the publishable web service. After entering the details in the form the details are entered in the database as shown in figure 3.11.

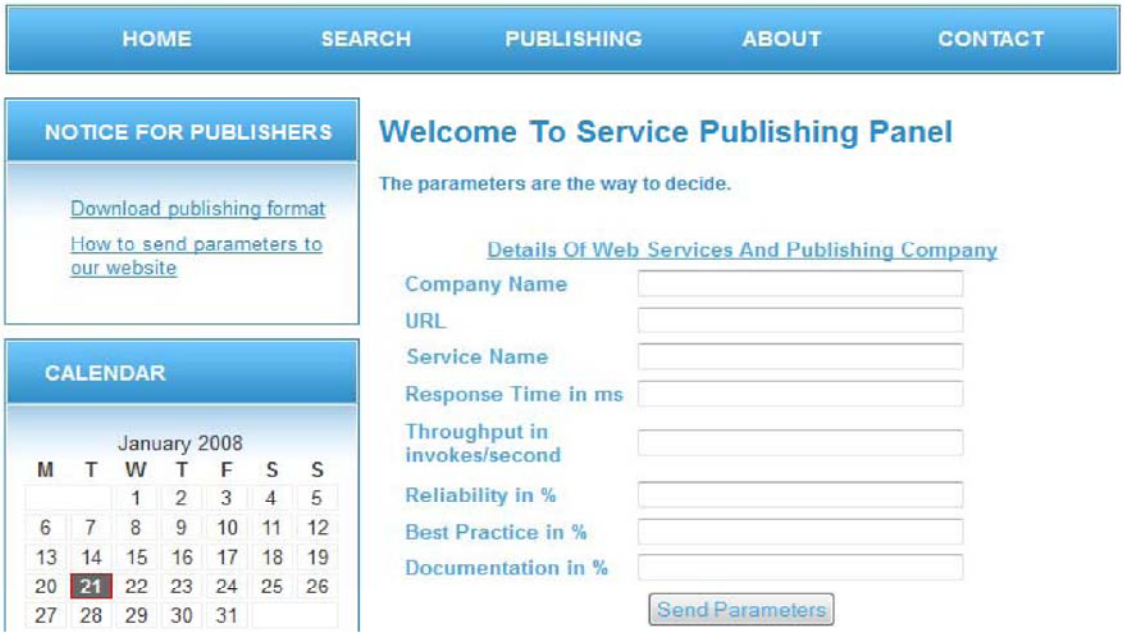


Figure 3.11: Snapshot of the service publishing panel

A message appears that the web service and its details are added in the database as shown in figure 3.12.

Figure 3.12: Snapshot of the submission of a web service in the service publishing panel

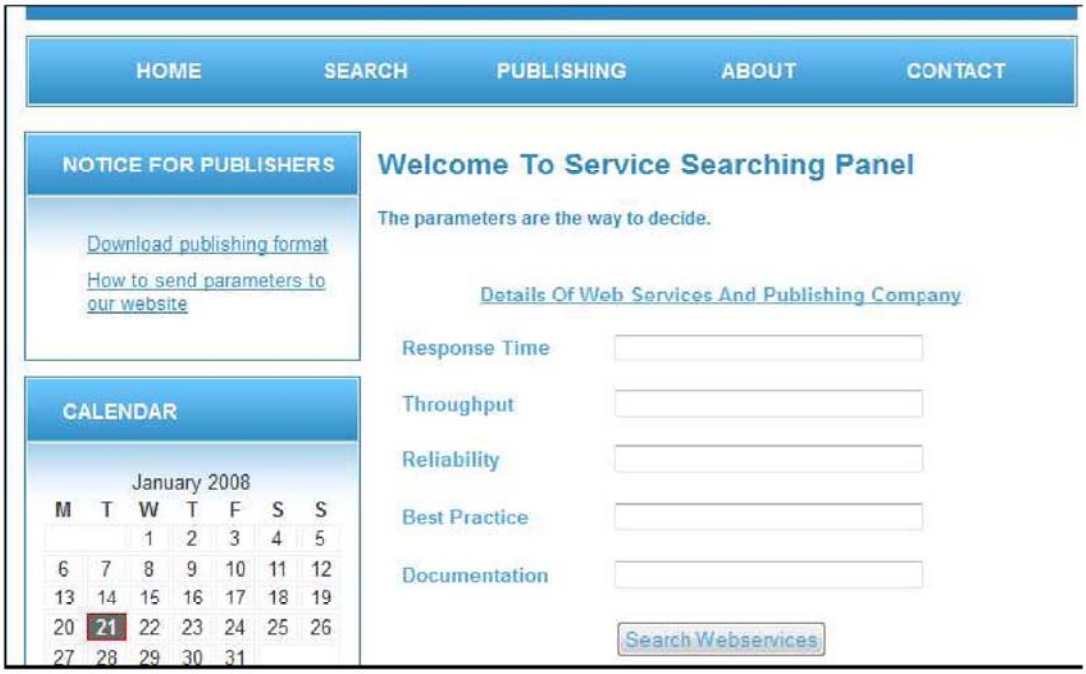


Figure 3.13: Snapshot of the submission of a web service in the service searching panel

If a service requester searches the web service, he will click on the search icon on the home page.

After entering the details of the web service the user is looking for, the list of relevant web service is provided as shown in figure 3.13 and figure 3.14 respectively.

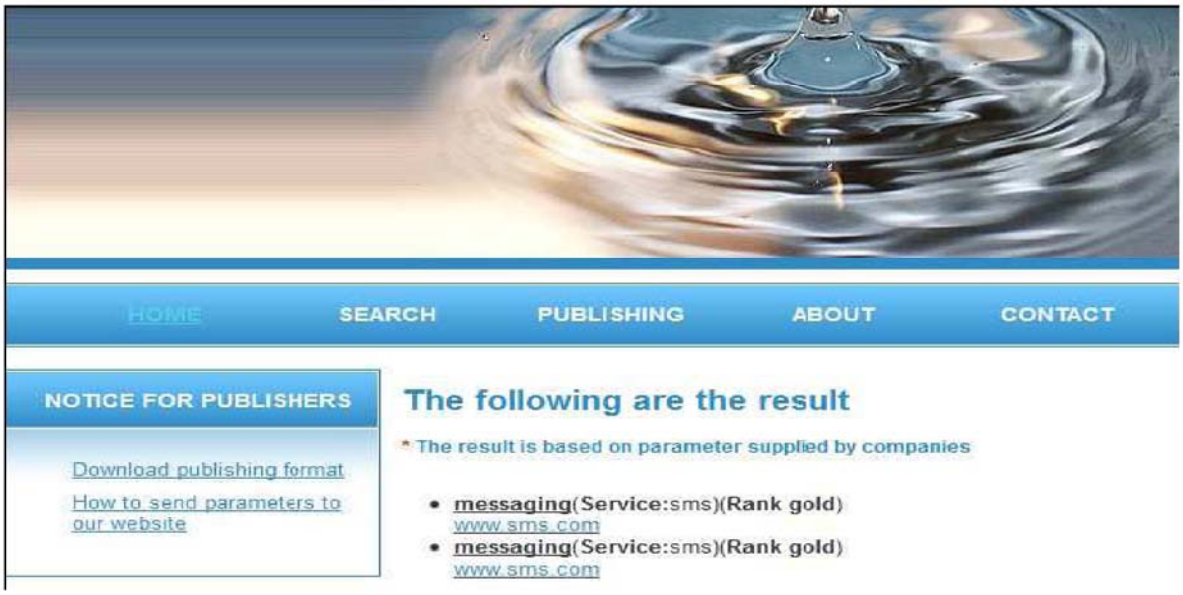


Figure 3.14: Snapshot of the submission of a web service in the service publishing panel

3.6 Results and Observations

After implementing the optimized business service directory, a check was done on the amount of heap memory consumed by the component and the time required to publish and search a service.

The figure 3.15 shows the heap memory consumption of the component. It can be seen that there is fluctuation in the heap memory consumed when the process in NetBeans IDE starts execution and later it depicts a constant consumption of memory i.e. in this case it consumes 10MB of memory. The figure shows that the memory consumption is not different in case of parsing 26 rules in comparison to 256 rules.

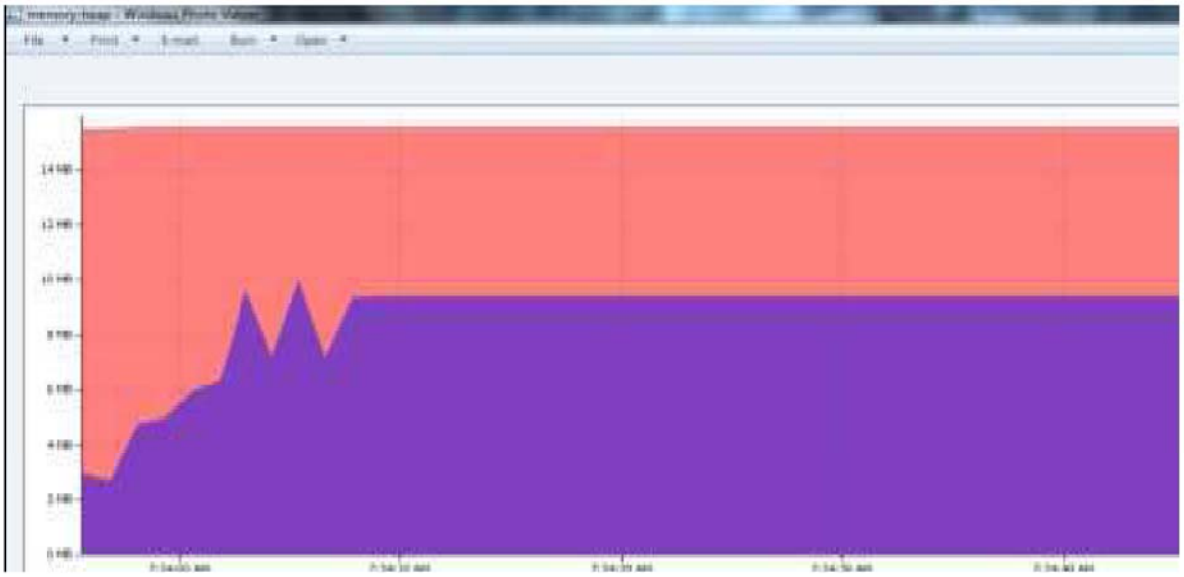


Figure 3.15: Snapshot of the heap memory consumption of the component

Figure 3.16 shows the time required to perform web publishing. The time required to process the request to publish the service is 122*10-3 sec. The snapshot in figure 3.17 shows the time required to process the request of web search is171*10-3msec, and to parse the xml rules time required is 39.9*10-3 sec.

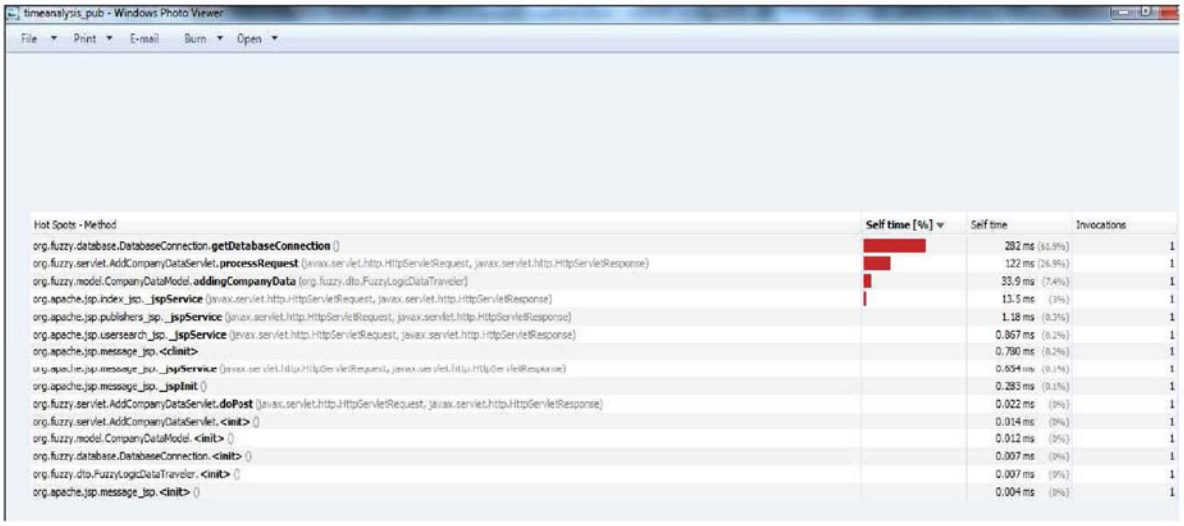


Figure 3.16: Snapshot of the time consumed to perform web publishing

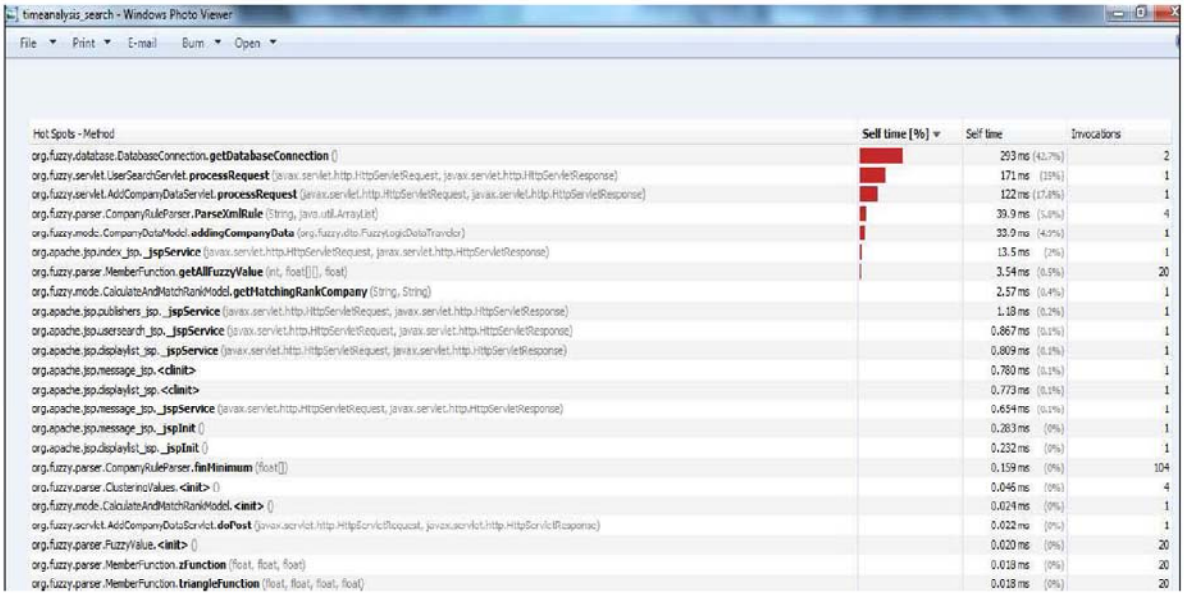


Figure 3.17: Snapshot of the time consumed to perform web searching

On carrying out the qualitative analysis of the technique described above to generate the rules, as apart of the research work in comparison to the technique which uses fuzzy logic the followingobservations are made:

1.The time required to search the web service in the database also called as seek time will be significantly less as it will be dependent on the number of rules required to trigger and the number of rules is less in the proposed technique as compared to implementing the BSD using fuzzy logic. The time complexity to parse the rules is O (N) where N is the no of rules.

Lesser the number of rules to trigger less will be the seek time. The seek time is also affected by the database size and the number of quality attributes considered qualifying the web service.

In the scenario considered and the related work, efforts were directed towards search for seven web services and it was found that the time required by the rule generation technique using PSO and fuzzy clustering is faster than the alternative technique which uses fuzzy logic [16].

The time was calculated by considering the time required to execute

ParseXmlRule(String xmlPath, ArrayList parameters) function in the project.

Table 3.4: Comparison of time consumed to parse 26 rules vs. 243 rules

services	Service 1	Service 2	Service 3	Service 4	Service 5	Service 6	Service 7
Time required to parse the 26 rules generated by Fuzzy clustering and PSO	187ms	263 ms	39.9 ms	222 ms	301 ms	322 ms	343 ms
Time required to parse the 243 rules generated by Fuzzy logic	693 ms	613 ms	309 ms	371 ms	480 ms	530 ms	574ms

2. The rules are successfully generated automatically using dataset thus making the system intelligent, in comparison to the technique using fuzzy logic [6] where the rules are to be entered by a human expert. Human intervention makes the system error prone and manual.

3. The rules are adaptive i.e., any change in the dataset or the ranking criteria will automatically be reflected in the rules and thus a new set of rules will be generated. The web services can be ranked according to the new rules.

4. The quality of rules is dependent on the training dataset; the data should be less overlapping and should have all varieties of output. The rules can still be generated using less number of entries in the dataset.

3.7 Summary

This chapter has proposed the optimized BSD. The proposed business service directory is automatic in nature as it a fuzzy expert system, whichwill rank the web services according to the rules generated by dataset. The PSO and fuzzy clustering reduces the rules. Considering the available data set, the number of rules is reduced from 243 (product of linguistic values of input and output variables) to 26. The lesser the number of rules, faster will be the processing

of ranking. The component is designed and can be used as a registry to publish and search the web services. The architecture is adaptive in nature as any change in QoS of a web service will change the rank of the web service. The proposed optimized service registry will enable one to develop a better B2B or a B2C kind of e-commerce application with agility. The service requester can compare among the list of web service and choose the appropriate service provider based on its requirements. Finally to summarize, the proposed model is better as it automatically monitors the rank of the web service using the generated rules.

The following chapters present the SOAP model (chapter IV) and secure dissemination (chapter V). Both the chapters in particular address the main issue of XML processing and security.

CHAPTER IV: A SOAP MODEL AGAINST REWRITING ATTACKS AND INSECURE CONVERSATION

As discussed in the second chapter service selection and XML processing and Security form the basis of our research work. Due to broad scope XML processing and security is further divided into two rewriting attacks and secure content based dissemination. This chapter is addresses rewriting attacks while the next chapter addresses the issue of secure content based dissemination.

4. 1 Background

The communication among these web services is established using SOAP messages which are based on XML. XML is the most relevant means to provide interoperatablity among various entities. When in network a XML file can be prone to hacking and unauthorized access, thus affecting the data integrity and confidentiality which are supposed to be important issues of communication. Data integrity and confidentiality can be breached by rewriting attack and insecure dissemination.

SOAP is defined as an enveloping protocol that is sometimes seen as a messaging protocol as well as a means of using functionality that is published by a remote application [4]. In other words, it acts as an adhesive to combine various heterogeneous and loosely coupled entities called as web services. The information passed in the message may represent either documents or Remote Procedure Calls (RPCs) that invoke specific procedures at the service provider. Figure 4.1 shows a typical structure of the SOAP message.

```
<?xml version="1.0"?>
<:Envelope
xmlns:="http://www.w3.org/2001/12/-envelope"
:encodingStyle="http://www.w3.org/2001/12/-encoding">
S<SOAPheaderheaderfor = "xyz">
<r:nodename>abc</r:nodename>
<r:nodeaddr>http://www.google.com</r:nodeaddr>
<r:hashvalue>hagsghghghghgfhgh12</r:hashvalue>
<r:hashvalue></r:hashvalue>
<:Header>
...
</:Header>

<:Body>
...
<:Fault>
```

Figure 4.1: SOAP Structure

It can be seen from figure 4.1 that the signed data object is referenced using a reference Uniform Resource Identifier (URI) within the XML Signature element, which is a child of the XML signature element. Thus the signed object is inside the XML Signature Element. The signed data object contains the XML Signature Element, which contains its signature, within it. Therefore the signed data object is the parent of its signature element [20]. This indirect referencing does not give any information regarding the actual location of the signed object.

Processing of the given SOAP message consists of two independent steps:

i. The recipient first searches for the referenced element and then computes the digest value over this element and compares it to the value given in the digest value. Later he verifies for the signature value.
ii. Next step is to execute the function defined in the SOAP body.

Therefore, the signed object can easily be relocated and the signature value still remains valid. This situation leads to rewriting attacks. Further, if the SOAP request passes through intermediaries en route to the destination web service, and if an authorized person has modified the content pretending someone else as the constructor, then there is no way to find out the end-to-end integrity [24].

4.2 Proposed Model

As evident from the related study and subsequent analysis of the work of past researchers have made us conclude that many researchers have worked towards securing SOAP message. We propose the SOAP model with three recommendations and ensure by implementing a case study that it will prevent rewriting attacks and have end to end integrity. The proposed model has following three recommendations

- Using timestamp in the message body and generate corresponding signature, the signature and timestamp needs to be encrypted by a shared key.
- Using value referencing for signature validation and message processing both, instead of using two different ways of referencing for both.
- Encrypting the signature and the timestamp with a shared key and put the encrypted value in the SOAP header. Later encrypt the whole SOAP body instead of sending an open SOAP Message in the network to prevent unauthorized access.

The figure 4.2 shows the structure of the SOAP message according to the proposed method.

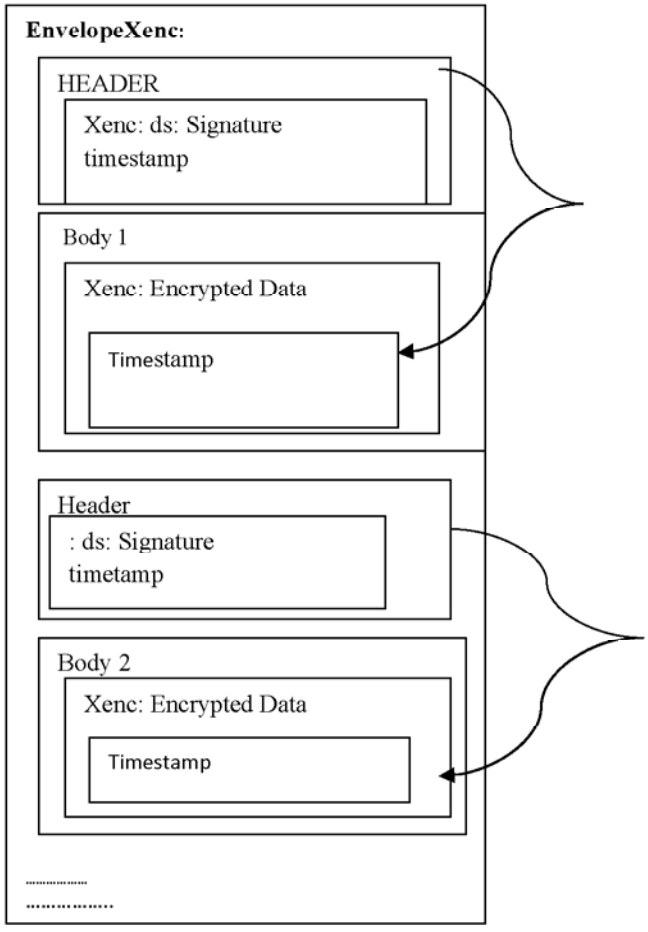


Figure 4.2: Proposed SOAP structure

The building blocks of the model includes: confidentiality, authentication, integrity and secure conversation.

4.2.1 Working of the Model

Each pair of entities in conversation possesses a shared key which is used to encrypt/decrypt the message according to the agreement. In general we assume that each entity in conversation is authentic and will not tamper the SOAP message. However message can be attacked by in-house and out-of-the-house (foreign) intruders. Out-of-the-house intruders, which are not the part of actual conversation, will not be able to decrypt the message as they will not have the shared key according to the model. They can at most cause total destruction of the message and make it unreadable for the entities in conversation. This attack can easily be traced down as the receiver will not be able to decrypt the message with the shared key.

An in-house intruder can be dangerous as it possesses the right key and can decrypt the message and modify it and send it further pretending that it is not the constructor. There may be two kinds of attacks which can be performed by in-house intruders:

1) Tampering the message created by different set of entity in communication

2) Re-locating the message or header under different fragment.

The case of tampering the message created by different set of entity in communication can be detected by signature validation. The message can be validated against the decrypted signature. It must be noted that different pairs of entities have different shared key. Due to this restriction, the in-house intruder will not be able to affect the signature (collectively containing the signature and timestamp) part of message generated by different pairs. If the key is demanded from the constructor, then the receiver will be responsible for the message he has forwarded to rest of the entities.

There are two purposes of introducing the timestamp in SOAP message:

i. To reduce the probability of false hit for any particular signature function used to generate the signature digest.
ii. To verify the message creation time as the in-house intruder can impose a false impression and try to re-arrange the order of the message which may be sometimes important. The order of message generation can be helpful as it depicts the correct path of message circulation.

To protect the message from second type of attack the *XPath* expression language can be used. This model is based on existing technologies like *XPATH* with Streaming API for XML (StAX). The reason behind using StAX over Document Object Model (DOM) [48] is that it's an offline data structure and thus has less memory consumption [49]. StAX is a *push parser* which refers to a programming model in which an XML parser pushes XML data to the client as the parser encounters elements in an XML infoset. Thus it enables to search from root to the leaf having the correct id instead of directly jumping to the node of given id. If re-location is done by the in-house intruder the search will fail and hence attack can be detected. This approach also avoids extra processing as the content can be copied under different fragment of SOAP message. The referencing to the node in this model is done using value hence the reference is called as *value referencing*.

The process of encryption and decryption of SOAP messages are specified below supported by the following symbolic notations:

E: The encryption algorithm used.

E^{-1}: The corresponding decryption.

G_s: The global key shared between all entities involved in conversation.

K_{AB}: Key shared between entity A and B.

T: the timestamp

SIG: The signature function

M_{AB}:Message shared between A and B

EM_{AB}: Encrypted message between A and B

SOAPB: The SOAP message body containing message

a. Web service workflow while sending a SOAP message

- Prepare the SOAP body M_{AB}
- Randomly insert Time stamp inside the message T_S.
- Now XML signature is generated out of the modified message.

$$S=SIG\ (M_{AB}+T)$$

- The next step would be to append the timestamp and signature inside the header of the message and encrypt it with a shared key K_{AB}. Whenever a receiver of the message wants to authenticate that the constructor, it will demand this Key K_{AB} from the constructor. While validating if the signature and the timestamp don't match that shows the message was tampered in the mid way by any authorized entity.

$$EM_{AB}=E(S+T, K_{AB})$$

- Later encrypt the entire SOAP message with a different shared key, called as group key Gs, This Key will be available to all the members of the group communication, who are authorized to append / read the message.

$$ESOAP = E(SOAPB + EM_{AB}, G_s)$$

- Send the prepared message through the network.

b. Web service workflow after receiving a SOAP Message

- Receive the SOAP message
- Decrypt the message with a shared key G_s in agreement, which is available with reliable agents only. Shared key ensures that the message can't be accessed by the unauthorized entity.

$$SOAPB' + EM_{AB}' = E^{-1}(ESOAP, G_s)$$

- If an authorized or trustworthy entity modifies the content then secure conversation can be ensured by validating the signature and the timestamp.
- If an authorized entity in the conversation changes some content and receiver requires checking its integrity. The receiver will request the constructor to send him the key K_{AB}. The signature and the timestamp is decrypted by the key and validated by the signature generated from the content. Signature as a sort of Hash function which can have collisions, timestamp can be then used as a collision resolver i.e., Timestamp will be validated only if there may be any false hit.

$$S' = SIG(M_{AB}')$$

If (S'=SIG(SOAPB'')) then message is verified

- If any other entity in the conversation wants to add information, it will append the information in the new body and its corresponding timestamp and signature is also maintained.

4.2.2 Model Implementation: Detection of XML Rewriting Attacks

For implementation the tools and technologies used included windows 7 operating system, JavaSE version 1.6, XML version 1.0, *NetBeans* version 7.1, Java Cryptography Architecture(JCA) API version 1.2, *Xpath* version 2.0, StAX version 2.0, JAX-WS 2.0 version, Signature was generated using SHA algorithm. In the example scenario Advanced

Encryption Standard (AES) with a key size of 128 bit (out of the various available encryption algorithms) is used to encrypt signature and timestamp. One can choose encryption algorithm depending on the difficulty level of security required.

While sending a SOAP message it is important to have an SOA environment that the should be free from recurring rewriting attack or insecure conversation, it's important to ensure that the SOAP message is sent and received without any tampering, even if the message has been tampered, there should be early detection. Consider a scenario, where a message shown below is sent by *WebservicesSOAPMessageCreator* to *Web servicesSOAPMessageAcceptor* as shown in figure 4.3.

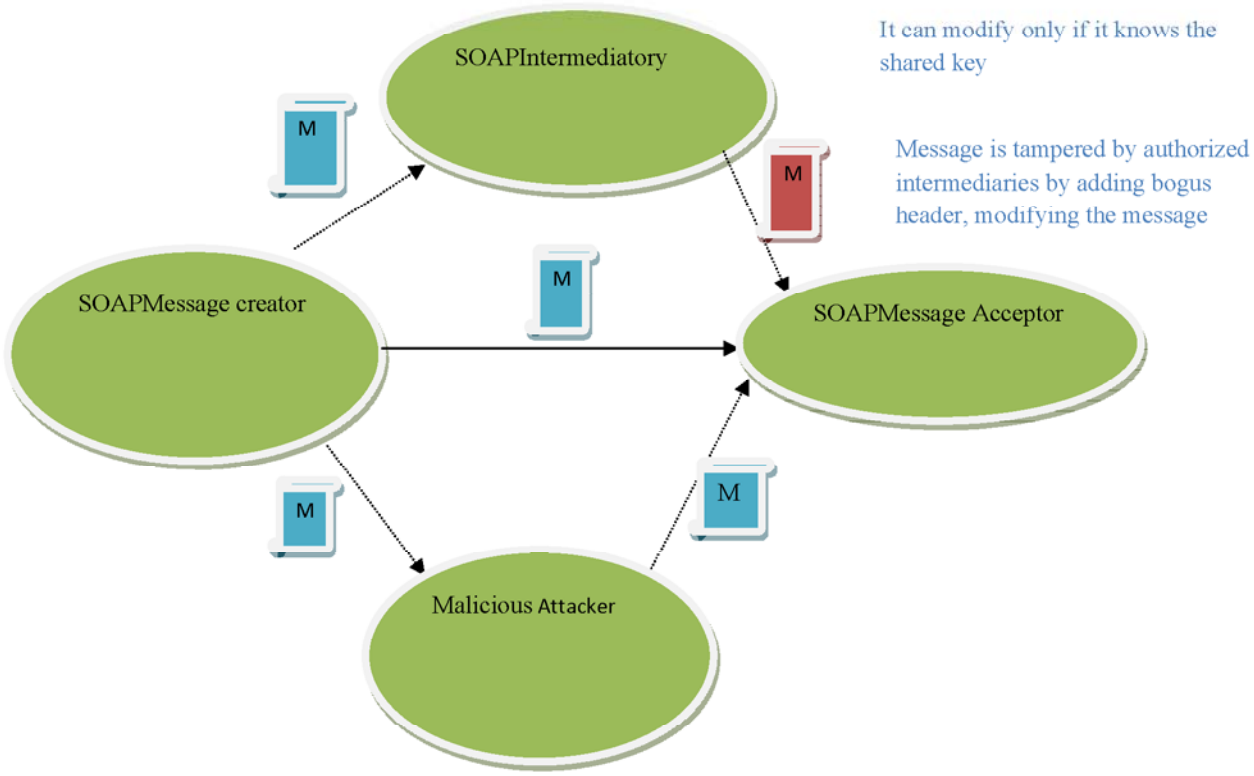
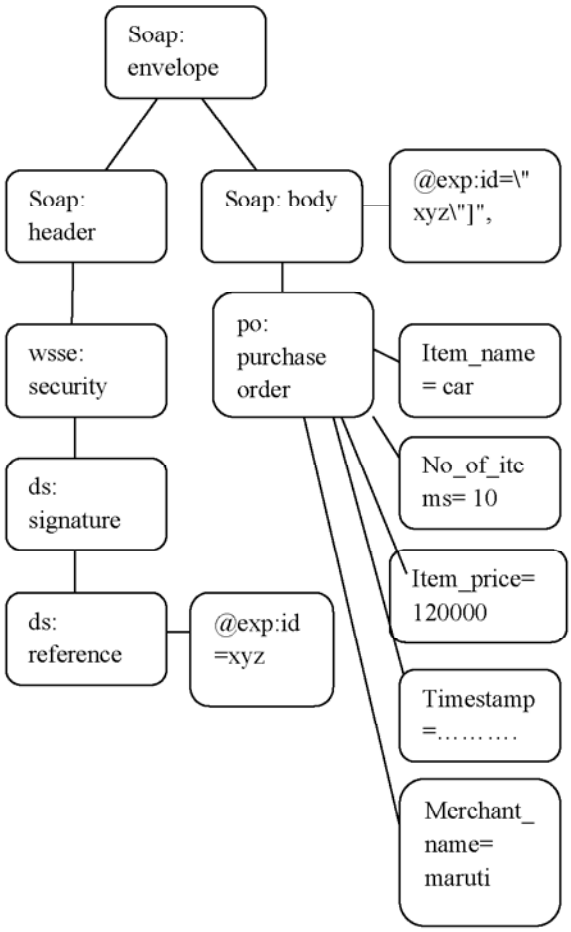


Figure 4.3: Depicting the attack scenario

The message is created using the steps mentioned in section 4.2.1 and sent over a network to the other web service. The message can't be tampered by the malicious attacker as it doesn't have the global key G_s. An in-house intruder can decrypt the message as it has the shared key G_s. The example scenario is shown in figure 4.3.

<?xml version="1.0" encoding="UTF-8" ?>
<SOAP-ENV:Envelopexmlns:SOAP-ENV="**http://schemas.xmlSOAP.org/SOAP/envelope/**">
<SOAP-ENV:Headerxmlns:exp="**http://SOAPexperiment.com**" exp:id="**xyz**">
<exp:hashValuexmlns:exp="**http://SOAPexperiment.com**">
<xenc:EncryptedDataxmlns:xenc="**http://www.w3.org/2001/04/xmlenc#**"Type="**http://www.w3.org/2001/04/xmlenc#Content**">
<xenc:EncryptionMethod Algorithm="**http://www.w3.org/2001/04/xmlenc#aes128-cbc**" />
-
<ds:KeyInfoxmlns:ds="**http://www.w3.org/2000/09/xmldsig#**">
-
<xenc:EncryptedKeyxmlns:xenc="**http://www.w3.org/2001/04/xmlenc#**">
<xenc:EncryptionMethod Algorithm="**http://www.w3.org/2001/04/xmlenc#kw-tripledes**" />
-<xenc:CipherData>
<xenc:CipherValue>**PaVSntYG8JzUwXeivEDosiKy8sUCU80NmuW+f27+XIc=**</xenc:CipherValue>
</xenc:CipherData>
</xenc:EncryptedKey>
</ds:KeyInfo>
-<xenc:CipherData>
<xenc:CipherValue>……</xenc:CipherValue>
</xenc:CipherData>
</xenc:EncryptedData>
</exp:hashValue>
</SOAP-ENV:Header>
<SOAP-ENV:Body id="**xyz**">
<po:Purchase-Orderxmlns:po="**http://SOAPexperiment.org/purchase**">
<item-details>
<item-name>**Car**</item-name>
<time-stamp>**Mon Apr 02 00:56:12 GMT+05:30 2012**</time-stamp>
<item-price>**120000**</item-price>
<number-of-item>**10**</number-of-item>
<merchant-name>**maruti**</merchant-name>
</item-details>
</po:Purchase-Order>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

(a)



(b)

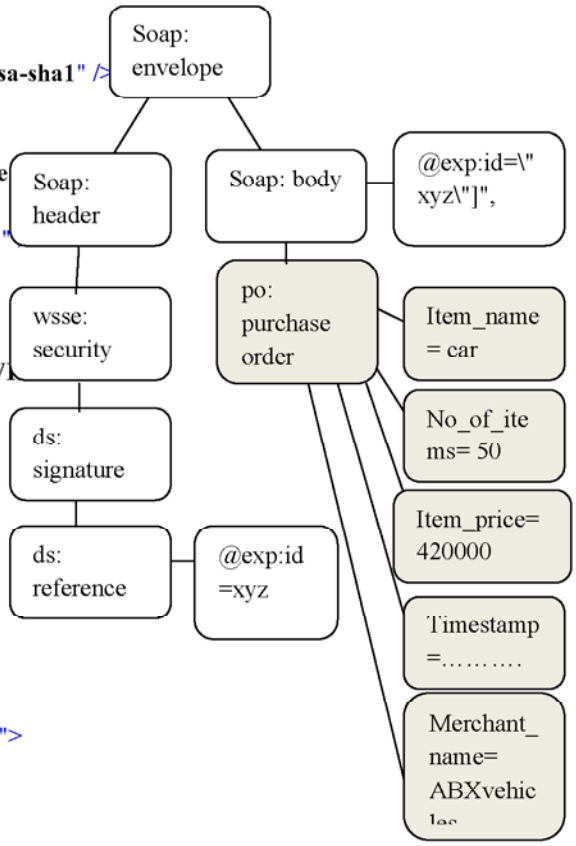
Figure 4.4: (a) Decrypted / original message SOAP message and (b) Represents its hierarchical diagram

Message after going through the network reaches at the receiving end, is then decrypted by the key G_s. The decrypted message is shown in figure 4.4.

If a trusted entity in the conversation breaks the trust and sends the message to some other after modifying it and pretends that he is just a sender not the constructor. The message can be validated whether it was modified in the transit, by validating against the signature and the timestamp as explained above.

```
<?xml version="1.0" encoding="UTF-8" ?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlSOAP.org/SOAP/envelope/">
<SOAP-ENV:Header>
<exp:hashValue xmlns:exp="http://SOAPexperiment.org/exp">
<Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
<SignedInfo>
<CanonicalizationMethodAlgorithm="http://www.w3.org/2001/10
ml-exc-c14n#WithComments" />
<SignatureMethodAlgorithm="http://www.w3.org/2000/09/xmldsig#dsa-sha1" />
<Reference URI="">
<Transforms>
<TransformAlgorithm="http://www.w3.org/2000/09/xmldsig#envelope
</Transforms>
<DigestMethodAlgorithm="http://www.w3.org/2000/09/xmldsig#sha1"
<DigestValue>PVlCxi6kV9tsk7PR3PJJtCwKuqw=</DigestValue>
</Reference>
</SignedInfo>
<SignatureValue>dhwi4ouKSEHxuLeYVtY2dg8NkzFgorhCHzwclW
GcAdYpQZ+gfPAg==</SignatureValue>
<KeyInfo>
<KeyValue>
<DSAKeyValue>……..</DSAKeyValue>
</KeyValue>
</KeyInfo>
</Signature>
</exp:hashValue>
</SOAP-ENV:Header>
<SOAP-ENV:Body id="xyz">
<po:Purchase-Order xmlns:po="http://SOAPexperiment.org/purchase">
<item-details>
<item-name>car</item-name>
<time-stamp>Wed Mar 07 11:17:00 GMT+05:30 2012</time-stamp>
<item-price>420000</item-price>
<number-of-item>50</number-of-item>
<murchant-name>ABX vehicles</murchant-name>
</item-details>
</po:Purchase-Order>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

(a)



(b)

Figure 4.5: (a) SOAP message with content tampered and (b) represents its hierarchical diagram

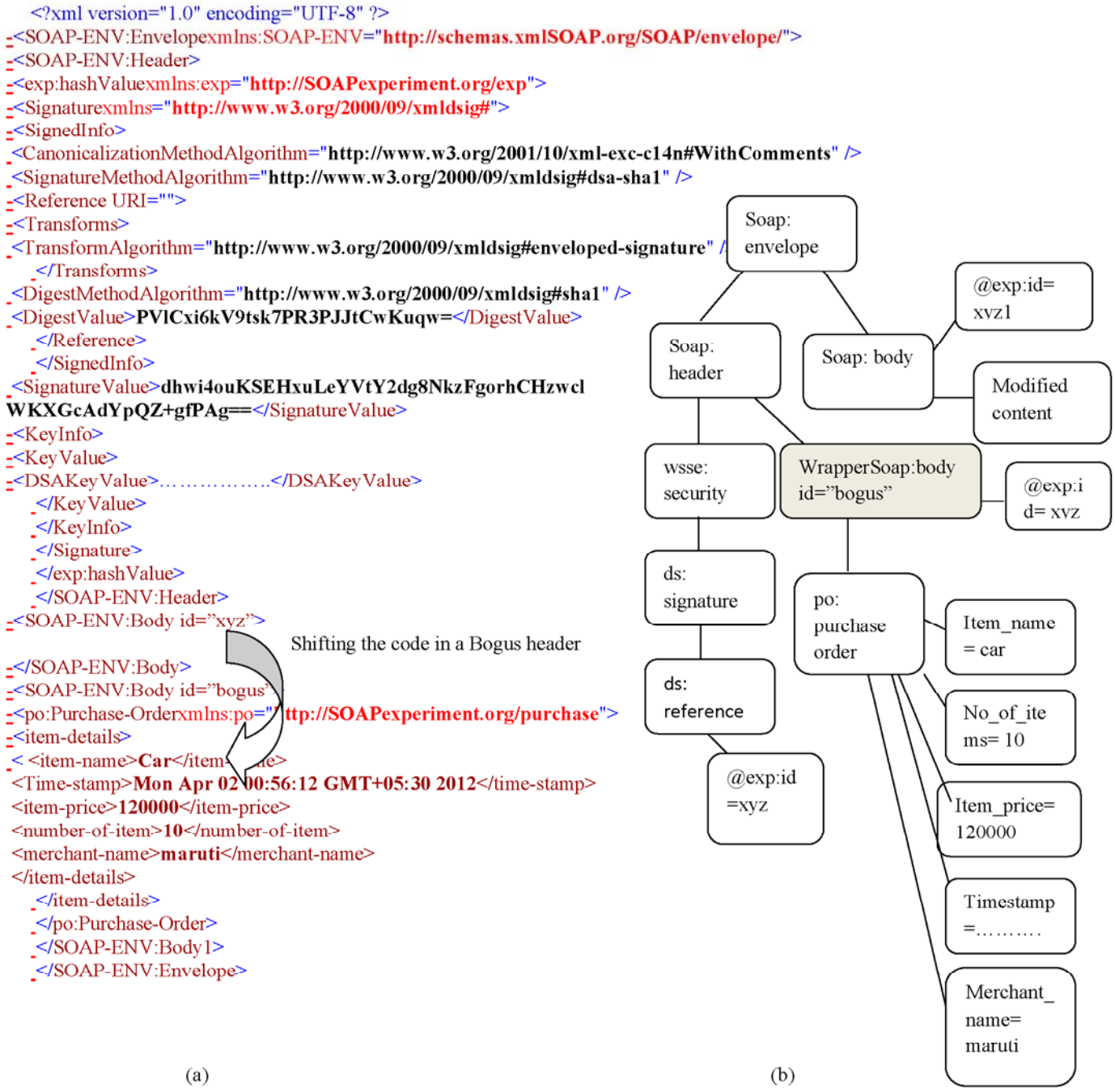


Figure 4.6: (a) SOAP content where the message is shifted to a bogus header (b) shows the hierarchical structure

Figure 4.5 and 4.6 shows the example of attack an authorized entity can perform. Fig 4.5 shows how the content can be tampered, but it will be detected as the signature will not be validated as per the algorithm. The attacker cannot change the signature as it is encrypted, and if it requires changing the signature, it has to ask for the key K_{AB}. Once it has obtained the key K_{AB} from *SOAPMessageCreator, it* will not be able to pretend as if it is not the constructor

Figure 4.6 shows how an authorized entity can shift the message to a bogus header. Even if the body is shifted to a bogus header, Instead of directly jumping to the node of given id, a search is made from root to the leaf having the correct id *SOAP-ENV:Envelope/SOAP-ENV:Header[@exp:id=\"xyz\"]"* as shown in Figure 4.7. The search will fail and hence attack can be detected. This approach also avoids extra processing if the content is copied under different fragment of SOAP message. Thus the model successfully detected rewriting attacks and established secure conversation

```
NodeList SOAPmhead=(NodeList)path.evaluate("/SOAP-ENV:Envelope/SOAP-ENV:Header[@exp:id=\"xyz\"]", document,
XPathConstants.NODESET);
DOMSignContextdomsc=new DOMSignContext(keyPair.getPrivate(),SOAPmhead.item(0));
XMLSignatureFactorysignatureFactory=XMLSignatureFactory.getInstance("DOM");
Reference reference=signatureFactory.newReference("#xyz", signatureFactory.newDigestMethod(DigestMethod.SHA1, null),
Collections.singletonList(signatureFactory.newTransform(Transform.ENVELOPED, (TransformParameterSpec)null)), null,
null);
SignedInfo
si=signatureFactory.newSignedInfo(signatureFactory.newCanonicalizationMethod(CanonicalizationMethod.INCLUSIVE_WITH
_
COMMENTS, (C14NMethodParameterSpec)null), signatureFactory.newSignatureMethod(SignatureMethod.DSA_SHA1, null),
Collections.singletonList(reference));
KeyInfoFactorykeyInfoFactory=signatureFactory.getKeyInfoFactory();
KeyValuekv=keyInfoFactory.newKeyValue(keyPair.getPublic());
KeyInfo keyInfo=keyInfoFactory.newKeyInfo(Collections.singletonList(kv));
```

Figure 4.7: Processing the SOAP message at the receiving end

4.3 Results and Observations: Performance Issues in Model Evaluation

After implementing the example scenario in section 4, performance of the SOAP model was analyzed as any signature and encryption operation on XML message requires considerable XML processing time which is directly proportionate to the size of the message [26].

The time required to construct the message = Time required generating signature + time required to encrypt the signature with K_{AB} + time required to encrypt the SOAP body with G_S.

The time required to process the SOAP message = Time required decrypting the SOAP body with G_S + Time required validate signature.

Incorporating all the three recommendations on the SOAP message would add overhead in terms of time required to construct the message at the sender head, time required to process the message in the receiving end.

Analysis of the effect of increase in SOAP size with respect to following criteria was done:

(1)The time required for encrypting the message for sending

(2)The time required for decrypting of the received message

(3)The time required for encrypting the signature

The snapshot of time profiling in *NetBeans* is shown in figure 4.8.

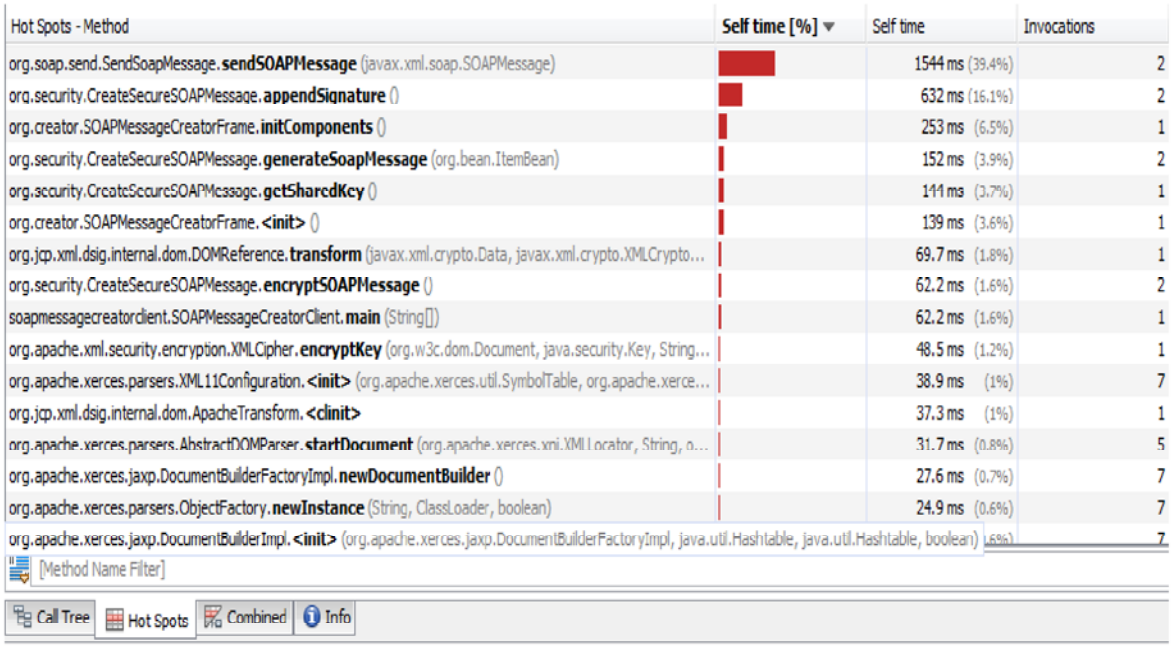


Figure 4.8: Snapshot of time profiling of SOAPcreator in *NetBeans*

To analyze the overhead of the encryption and decryption, we plotted a graph as shown in figure 5.9. The graph depicted the plot of time required for message encryption and decryption versus the number of tags in the SOAP message. The time required to encrypt a SOAP message with 5 tags was 102 milliseconds and as the number of tags grew to a count of 50 the time required to decrypt the message was 124 milliseconds. The graph clearly shows that the increase in y-axis is too less in comparison to the corresponding increment in the y-axis. The percentage of growth was .04%. The red line in the graph shows time taken to decrypt the message with key G_s. It was observed that the time required to decrypt a message

of 5 tags was 119 milliseconds and as the SOAP message size grew to 50 tags the time take to decrypt the message was 183 milliseconds. In this case also it was seen that the overhead of decrypting doesn't subsequently increase with the number of tags in the SOAP message. The percentage of growth in the time with respect to the number of tags was 0.14%.

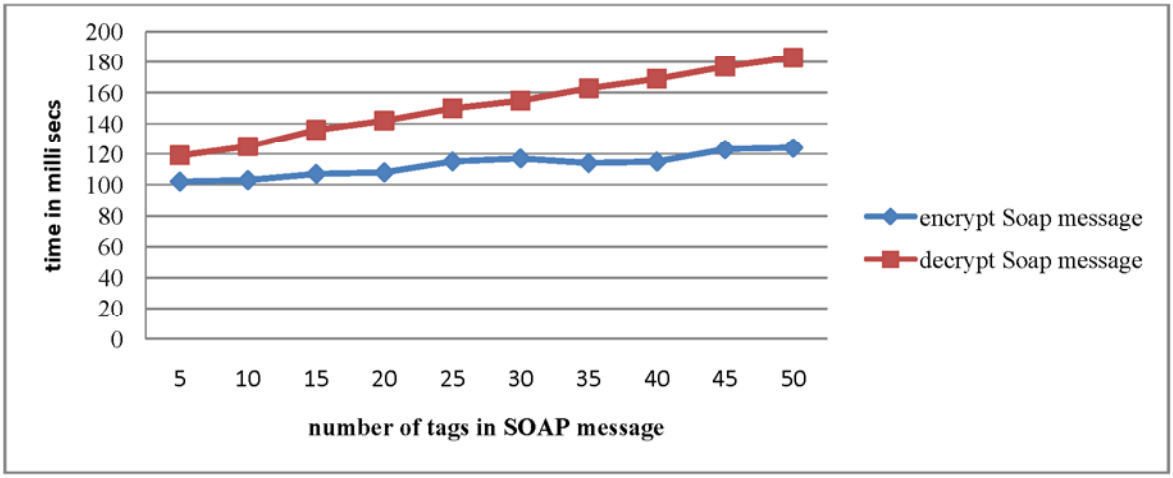


Figure 4.9: Encryption of the message for sending with respect to number of tags in the SOAP message

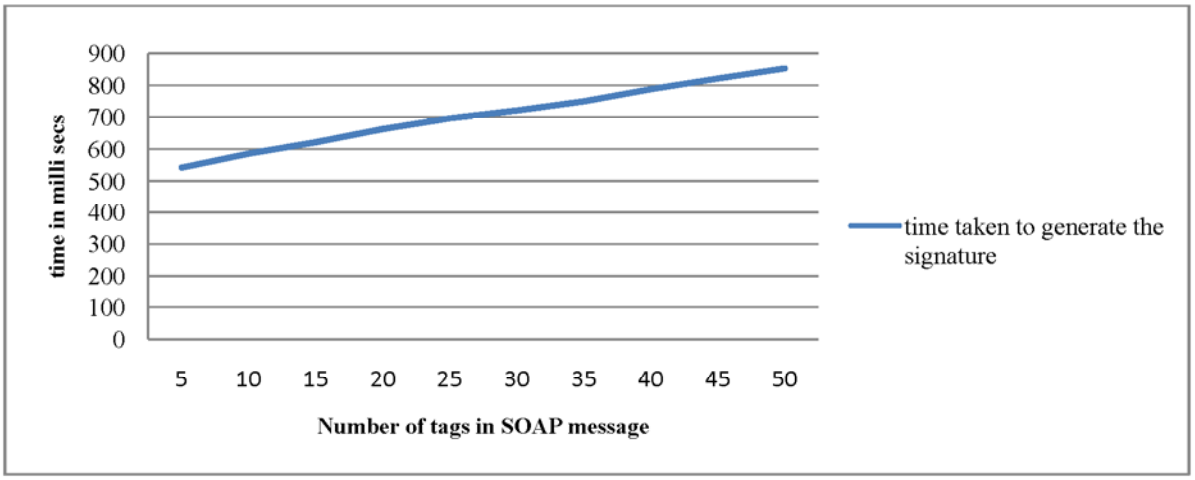


Figure 4.10: Time required to encrypting the signature with respect to number of tags in the SOAP message

As the recommendations in the model also contained generation of signature, it was observed that how the time taken to generate the signature varies as number of tags increases. The graph in figure 4.10 shows the time required for encrypting the signature with respect to number of tags in the SOAP message. Time required for Signature generation will depend on the key generation algorithm like Data Encryption Standard (DES) or AES.

It depends on the programmer which key generation algorithm he/ she wants to use depending on the difficulty level of security required. The time required to generate the signature for a message with 5 tags was 542 milliseconds, while the time taken to generate the signature for a message with 50 tags was 788 milliseconds. It can be seen that the percentage of growth is 0.5%. Thus it can be inferred that the proposed SOAP model doesn't have much overhead.

We have compared computing time for generation of encrypting the whole SOAP messages and plain SOAP message generation w.r.to number of tags, it was observed that the average overhead of sticking to the proposed model was 2%. The graph in figure 4.11 depicts the same. Thus it can be inferred that the proposed SOAP model has significantly less overhead.

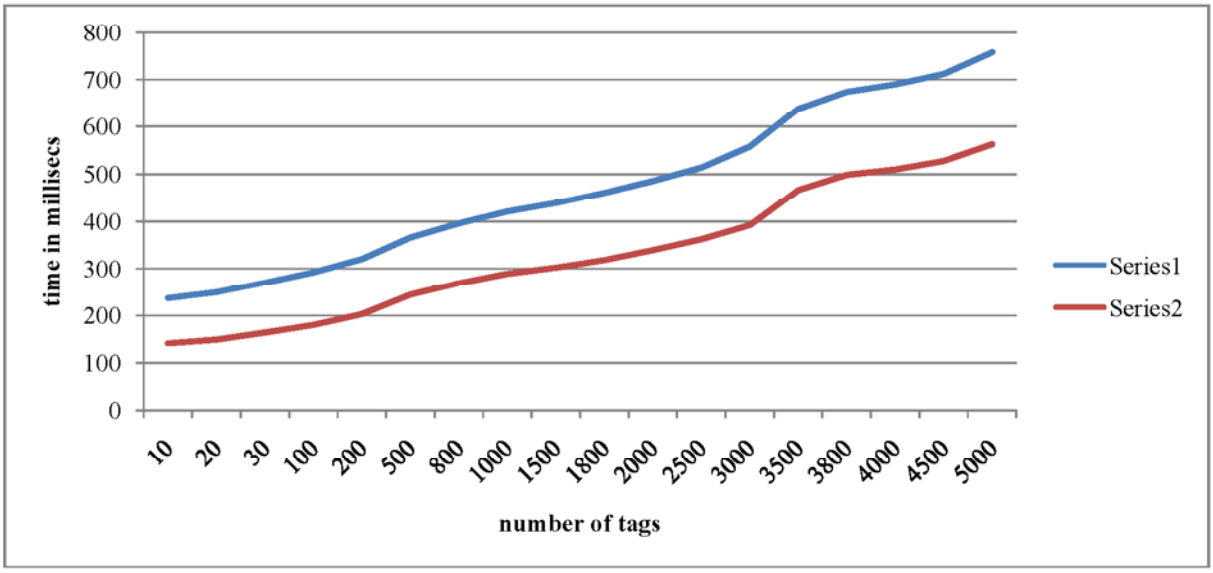


Figure 4.11: Computing time required to create an plain SOAP message (series 1) and encrypted message (series 2) with respect to number of tags in the SOAP message

4.3.1 Comparison of the Proposed Model with the Earlier Models

[21] Has proposed *FastXPath*, which is not flexible as it limits the abilities of defining a signature reference. The proposed model is based on existing technologies like *XPATH* with StAX. The reason behind using StAX is to enable search from root to the leaf having the correct id instead of directly jumping to the node of given id. If re-location is done by the in-house intruder the search will fail and hence attack can be detected. This approach also avoids extra processing as the content can be copied under different fragment of SOAP message. The referencing to the node in this model is done using value hence the reference is called as *value referencing*. Thus it does not limit the abilities of defining signature reference

[22] Recommended using depth information, parent information and using *Id* attribute to uniquely identifying the parent. As the depth information can also be tampered by the malicious attacker. We are not attaching the extra information, we are just attaching timestamp and embedding it into SOAP body and then encrypting the whole SOAP body. Thus the malicious attacker cannot access the added information

[23] has an disadvantage that it doesn't comply with the schema of the WS* standards, our model sticks to the basic structure of SOAP model mentioned in WS standards.

[24] has the disadvantage that the process will be slow as signature generation is a slow process. Our model has no nested signature as signature generation is a time taking process; we have lesser levels of signature

[25] The trade off is that the context is to be generated and stored in the reference element of the signature in header section before signing the message. We are not involving any extra step of generating context, thus it will have less time taking than the method suggested in [25].

4.4 Summary

This chapter attempts to highlight that rewriting attack is the price that one has to pay for the robustness, extensibility and flexibility of web service specifications and also for the separation of concerns in the web services frameworks [49]. It was to address these challenges of rewriting attacks and secure conversation that a SOAP messages transmission faces in enterprise web service applications that model aimed to achieve the properties of simplicity and light transmission has been developed.

Further in this chapter the working of the model is based on three possible recommendations namely, using shared key for encrypting timestamp in the message body for generating corresponding signature. Second recommendation is using value referencing both for signature validation and message processing. Final recommendation is encrypting the whole SOAP body instead of sending an open SOAP Message in the network to prevent unauthorized access.

The following chapter continues this with the work on secure dissemination i.e., the third and the last component of the research work.

CHAPTER 5: SECURE CONTENT BASED DISSEMINATION OF XML CONTENT

As mentioned in the previous chapter secure dissemination of an XML file is one of the techniques to ensure data integrity and confidentiality. This chapter proposes a secure dissemination technique such that extraneous data not meant for a consumer should be inaccessible. The information should not be accessible to authorized consumer, if it is not meant for him as flow of extraneous data may leak information.

5.1 Background

Let us consider a scenario where large documents at the source end called as producer, can be accessed by large number of service requesters called as consumers. The consumer should be able to view only relevant data which he has been authorized to view. Even a legitimate consumer can exploit the knowledge of context from the data elements it has access to [29]. These access policies should be specified by the producer. Flow of extraneous data to a consumer may lead to information leakage. In particular the extraneous data is prone to off-line dictionary attacks even by legitimate consumer that can exploit contextual knowledge from the data elements it has to access [30].

Security of the data is an important issue as having better QOS [2] is all in vain if data integrity cannot be ensured. The problem scenario of secure dissemination is represented in figure 5.1.

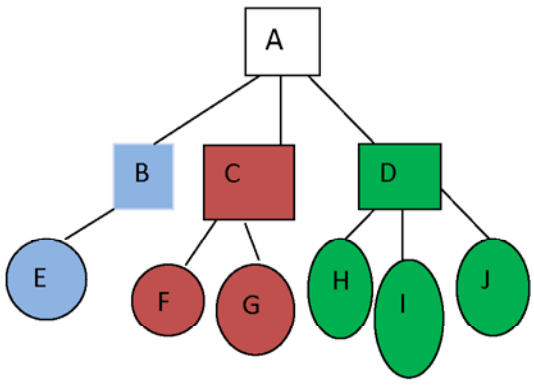


Figure 5.1: Hierarchical structure of XML tags

The n-ary tree T represents the tree structure of a XML generated by the producer or the service provider. The access policy for the scenario shown in figure 5.1 depicts that the nodes in blue are requested by consumer 1, red nodes and green can be accessed by consumer 2 and consumer 3 respectively.

The various technologies used and details of their features which have been used is described in section 5.1.1 and 5.1.2.

5.1.1 XML

XML [50] is a declarative and narrative language and need parser of type DOM to fetch out useful information described therein at processing end. DOM [51] structure is a composed *element*. An element contains a portion of the document delimited by two *tags*: the *start tag* at the beginning of the element, with the form <*tag-name*>, and the *end tag*, at the end of the element, with the form </*tag-name*>, where *tag-name* indicates the type of recommended data. The structure looks like as shown as in figure 5.2.

```
<documcnt>
 <last_updated>July 28, 2012</last_updated>
  <copyright>ABC company</copyright>
 <maintainer email="ras@juitcom"
        url="http://www.juit.com/">
  <name>
   <first_name>marko</first_name>
            <middle_name>D</middle_name>
  <last_name>Harold</last_name>
 </name>
          </maintainer>
       </ document>
```

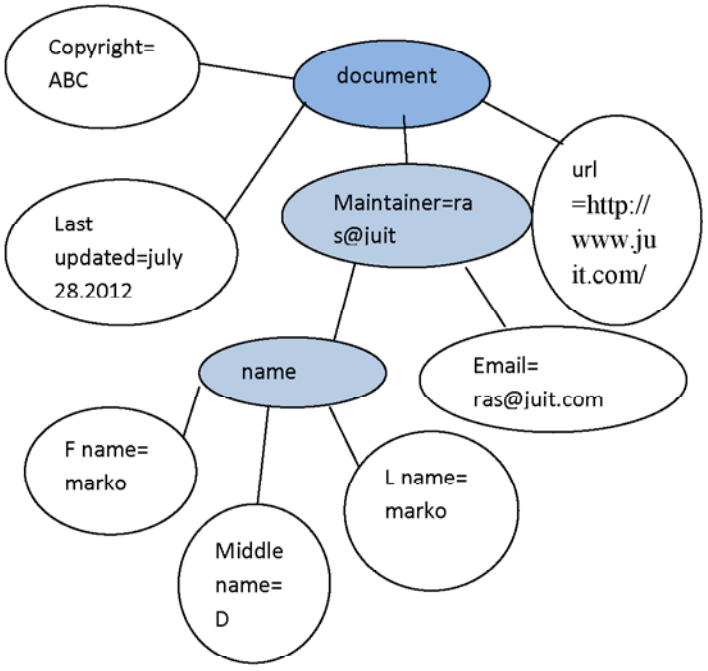


Figure 5.2: An XML document and its corresponding tree structure.

An XML document has an underlying tree structure. The nodes of the tree represents elements their attributes, and edges represent relationships between them [2]. A tree can be represented in the computer memory in many ways.

One of the sequential techniques and a compact representation of tree structure is to represent tree in memory using linear list [52]. It is the most common representation of the tree states that the nodes can be stored by consecutive addressing.

The tree in figure 5.3 can be represented in the memory as (A (B (E), C (F, G), D (H, I, J))

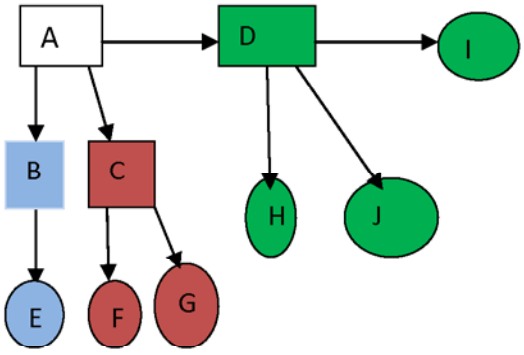


Figure 5.3: Linear list representation of the tree in memory

The root A will have a link to all of its children's from left to right; the children's will further link to their children. Thus if the producer want to transmit the data which consumer 3 has subscribed for, only the address of D should be known to the consumer and the whole subtree can be traced easily. Thus any subtree is accessible if a consumer is able to locate the address of the root of the subtree. We are trying to exploit the above property of the tree, for securely disseminating the tree.

To ensure security of data, the linear list can further be encoded in a DNA (Deoxyribonucleic acid) strand [54]. The properties of DNA are its compact nature and simplicity in implementation which makes it an ideal choice to encrypt the tree structure of XML data. DNA also provides high complexity at decoding end without right information.

5.1.2 DNA

DNA is a double stranded molecule as shown in figure 5.4. The building blocks of DNA double strand are the variant combinations of the four nitrogenous bases adenine (A), thymine (T), guanine (G) and cytosine (C). Two strands of DNA can form (under suitable conditions) a double strand if the respective bases are Watson-Crick complements of each other - A matches with T and C matches with G, also 3' end matches with 5'end [53].

The main idea of computing with DNA is to encode data in a DNA strand form in order to simulate arithmetical and logical operations using Watson crick complement [54]. It's like we are working with a Quaternary means 4-ary number system. High information density of DNA molecules and massive parallelism involved in the DNA reactions make DNA computing a powerful tool. DNA is a good medium for data hiding because of its great length and high randomness. DNA has the attribute of simplicity and compactness identified to choose a security protocol.

DNA cryptography has been proposed by Gehani et al. [55], Kartalopoulos [56] and Tanaka et al. [57] as a new born cryptography field. It's also called as DNA stenography, where we tend to hide data in a DNA strand with compactness and simplicity.

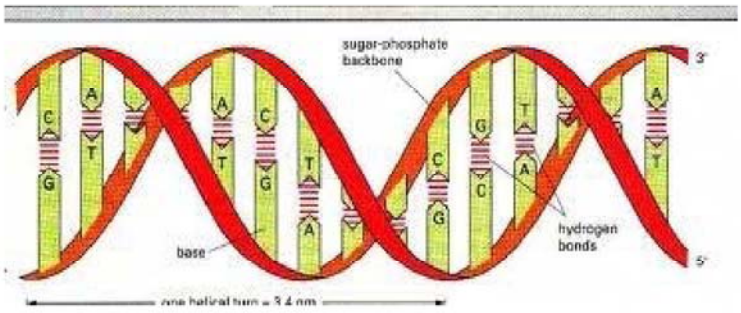


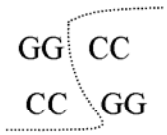
Figure 5.4: The double helical structure of DNA [58]

5.1.3 Restriction Enzymes

Restriction enzymes are molecular scissors that cut DNA into fragments at specific sites in their sequence. The enzymes degrade the foreign DNA by cutting the area that contains specific sequences of nucleotides. Since discovering the function of these enzymes, molecular biologists

have isolated them from a variety of single-celled organisms for cutting DNA into fragments [59].

HaeIII, isolated from Haemophilus aegyptius, is an example of a restriction enzyme. Its recognition sequence is:



Where ever in the DNA strand it will find the sequence shown above it will cut the strand between G and C.

There is a large set of database of available restriction enzyme [60, 61]. The RBASE database registers 4000 restriction enzymes. We are not focusing on the scissor property of the enzyme, but are concerned about their property of identifying the recognition site as shown in above example. This property of the restriction enzyme is used to specify the access policy.

The idea is to assign a Starting Restriction Enzyme (SRE) and Ending Restriction Enzyme (ERE) to a specific consumer so that when a data is prefixed and suffixed by the respective enzyme then the corresponding consumer can be located. We have encountered that in the real world there are 4000 types of restriction enzymes, this will limit the number of combinations to 4000(SRE) * 4000(ERE). Hence we have not picked up real restriction enzymes but our SRE and ERE consists of sequence of bits {A,T,C,G} of length n, where $n \epsilon \{ 1,2,3, \ldots \ldots \}$.

5.2 Proposed Secure Dissemination Technique

The technique consists of three things, three actors namely

i. 1 Producer, who prepares an encrypted XML document to transmit through the network

ii. *N* number of Consumer who has to read the message from the XML document

iii. A multicast dissemination interface, which act as disseminators which customizes the access policies for each customer by assigning a flag to the customer, such that the customer can extract the message meant for him through the XML file and decode it to get the message.

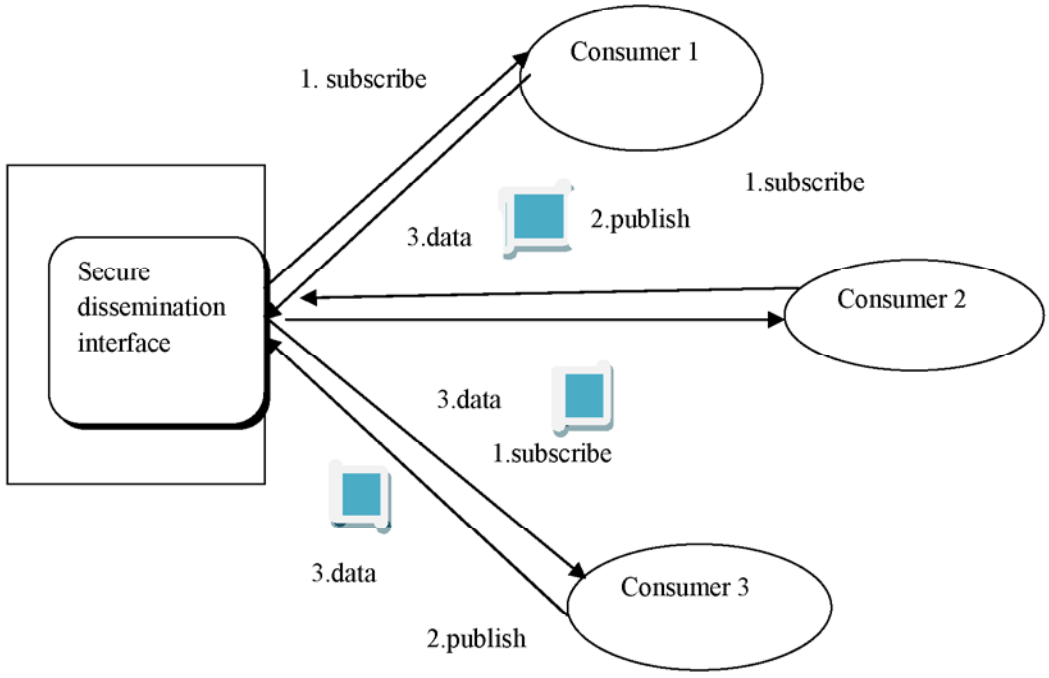


Figure 5.5: The architecture diagram of the secure dissemination interface

The secure dissemination technique is implemented as multicast dissemination interface which is built dynamically and asynchronously using publish / subscribe methodology as shown in figure 5.5. Once the consumer subscribes with the interface, it will be assigned a SRE and ERE. The consumer will subscribe with secure dissemination interface for the service which the service provider has published. The data is prepared according to the technique and sent to the consumer such that there is no information leak.

A temporary XML document is prepared by prefixing and postfixing the XML tags which the consumer is authorized as shown in figure 5.6. If a consumer is allowed to visit a top level tag then he/she is eligible to see all the child tags. If there are more than one subscriber of a node, for example x and y has subscribed for the root node, then any one pair of the SRE and ERE is used for annotation and the other consumer is informed about the new pair of SRE and ERE.

```
<document>
 <last_updated for="consumer3">July 28, 2012</last_updated>
 <copyright for="consumer2">ABC company</copyright>
 <maintainer email="ras@juitcom"
         url="http://www.juit.com/">
  <name for="consumer1">
   <first_name>marko</first_name>
   <middle_name>D</middle_name>
   <last_name>Harold</last_name>
  </name>
 </maintainer>
</ document>
```

Figure 5.6: The prepared temporary XML file

The interface starts processing this temporary XML document and later provides the following two options:

a. Produce one single file to send to all clients

In this case the interface chooses as many SRE and ERE as there are "for" attribute in the temporary XML file. Now before the start tag of each tag having "for" attribute the software will append information.

b. Produces separate file for each set of client

The interfaces will produce as many encoded file as there are "for" attribute in the temporary file. In this example there will be three output files. Each encoding is done with different set of SRE and ERE. The interface will give the name of SRE and ERE used for that particular file.

At the producer end the procedural flow involves:

i. It prepares a temporary XML file to be transported to the consumer by prefixing the data with a tag called as 'for consumer ___" .
ii. Later the temporary file is picked and where ever a 'for ' tag is located file, it's replaced by SRE and the data is suffixed by ERE of the concerned consumer number.
iii. The producer encrypts the content of all the XML elements in a DNA strand.
iv. Later transports it to various consumers.

At the consumer end when the data is received, the procedural flow involves:

i. The consumer receives the encrypted file
ii. Later it searches for the data that will fall between his specifies restriction enzymes.
iii. Consumer obtains data access to Root node and children address
iv. Data of the child nodes can be extracted from the addresses obtained from step ii.
v. This extraction continues till all the leaf nodes are obtained.

5.3 Working of the Proposed Secure Dissemination Technique

The content of the node in XML contains attributes of an XML element. That content is encoded in a way that will avoid malicious attacks. As stated in the introduction DNA encryption has been chosen for encoding the data. The DNA molecule is composed of four basic groups of A, C, G and T. Thus each XML node's content is treated as a text; each character of the text is parsed and encoded using computer mapped character encoding like UTF, ASCII.

5.3.1 XML File Encryption in a DNA strand

Here, we illustrate encoding for the character 'A'. The ASCII value is considered and its equivalent binary number equivalent is generated.

The procedure is outlined below:

ASCII value is considered and its equivalent binary no is generated. Let us take an example

A -> its ASCII value is 65

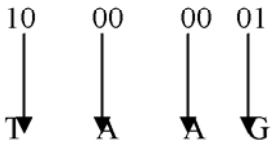
We now convert the ASCII value into its equivalent binary no is 100001. This binary number is the encoded in the form of DNA strand. As we know the DNA strand is composed of (A+ C+T+G)*, thus possible patterns for this encoding format are 4! =24. According to Watson-Crick complementarily rule [62, 63] nucleotide base A is complement to T and C is complement to G. Take DNA digital coding into account, it should reflect the biological characteristics of 4 nucleotide bases, the complementary rule that (~0) =1, and (~1=0) is proposed in this DNA digital coding. According to the complementary rule, the complement of 0(00) is 3(11) and 1(01) is 2(10) and vice versa. So among these 24 patterns, only 8 kinds of patterns (0123/CTAG, 0123/CATG, 0123/GTAC, 0123/GATC, 0123/TCGA, 0123/TGCA, 0123/ACGT, and 0123/AGCT) which are topologically identical fit the complementary rule of the nucleotide bases [64].

Among the eight patterns, we have chosen a representation as shown in table 5.1 to explain the method of encryption:

Table 5.1: Representation to map DNA nitrogenous bases to Binary number

Binary number	DNA encoding
00	A
01	C
10	T
11	G

Using the table 5.1 now the encoded DNA strand looks like the following:



Thus the nucleotide TAAG is the encrypted form of A.

Following the above explained procedure of encryption the root node in the figure 5.2 "document" is encoded:

01000100 01001111 01000011 01010101 01001101 01000101 01001110 01010100

CACACAGGCAAGCCCCCAGCCACCCAGTCCCA

Thus after encryption every data of node will look like: $\sum$ {A, T, C, G}.

To add an additional level of encryption, we recommend XORing the data with a key. Each character can be XORed with an 8 bit key, which will be known to all the consumers. An out house intruder has to try 2^8 apart from knowing SRE and ERE to decode a data.

The same word "document " if XORed by a key 10011011 becomes

Original form	01000100	01001111	01000011	01010101	01001101	01000101	01001110	01010100
Key	10011011	10011011	10011011	10011011	10011011	10011011	10011011	10011011
XOR	11011111	11010100	10110001	1001110 1	10101101	1011110	11010101	11001111

The DNA strand after encryption now appears to be

GCGGGCCA GCTAGATAGCCAGCGT GCAC GAGG

5.3.2 Assigning a Starting and Restriction Enzyme to Each Consumer

The property of restriction enzyme to identify the recognition site is used to specify the access policy. Thus two restriction enzymes are randomly assigned to each consumer subscribed for the XML data. This restriction enzyme is prefixed and suffixed to data as an annotation specifying that which consumer is authorized to access the data. One which is prefixed is called as SRE and

other one is called as ERE. Considering figure 5.1, there are three consumers subscribed to the document. x, y, z are the name assigned to them and their respective SRE and ERE are

Table 5.2. SRE and ERE assigned to the various consumer

Name of the consumer	SRE		ERE	
	Name	Recognition site	Name	Recognition site
x	*Eco*RI-	GAATTC	*Eco*RII	CCWGG
y	SmaI	CCCGGG	Sau3A	GATC
z	EcoRV	GATATC	KpnI	GGTACC

SRE and ERE of consumer x and y are mathematically represented as $<SRE_x, ERE_x>$ and $<SRE_y, ERE_y>$ respectively. When a data is hidden in a DNA strand, it's actually annoted with the consumer's SRE and ERE to signify that consumer x has the authorization to access the node and the subtree. If consumer x is authorized to access the root node then the strand appears to be

GAATTC **GCGGGCCA GCTAGATAGCCAGCGT GCAC GAGG** CCWGG

If there are more than one subscriber of a node, for example x and y has subscribed for the root node, then any one pair of the SRE and ERE is used for annotation and the other consumer is informed about the new pair of SRE and ERE.

5.3.3 Scattering of Data in the Garbage File

A garbage file F can be defined as a large text file consisting of random combinations of A, C, T, G. DNA cryptography has an advantage that it provides four options for each bit as it in Quaternary. Thus the numbers of combinations are 4 for each bit.

Figure 5.7 shows a graphical representation of number of bits versus number of combinations in terms of exponentiation. It can be seen that the values of binary is lower than quaternary. Table 5.3 shows the table of number of combinations versus number of bits for both quaternary and binary number system.

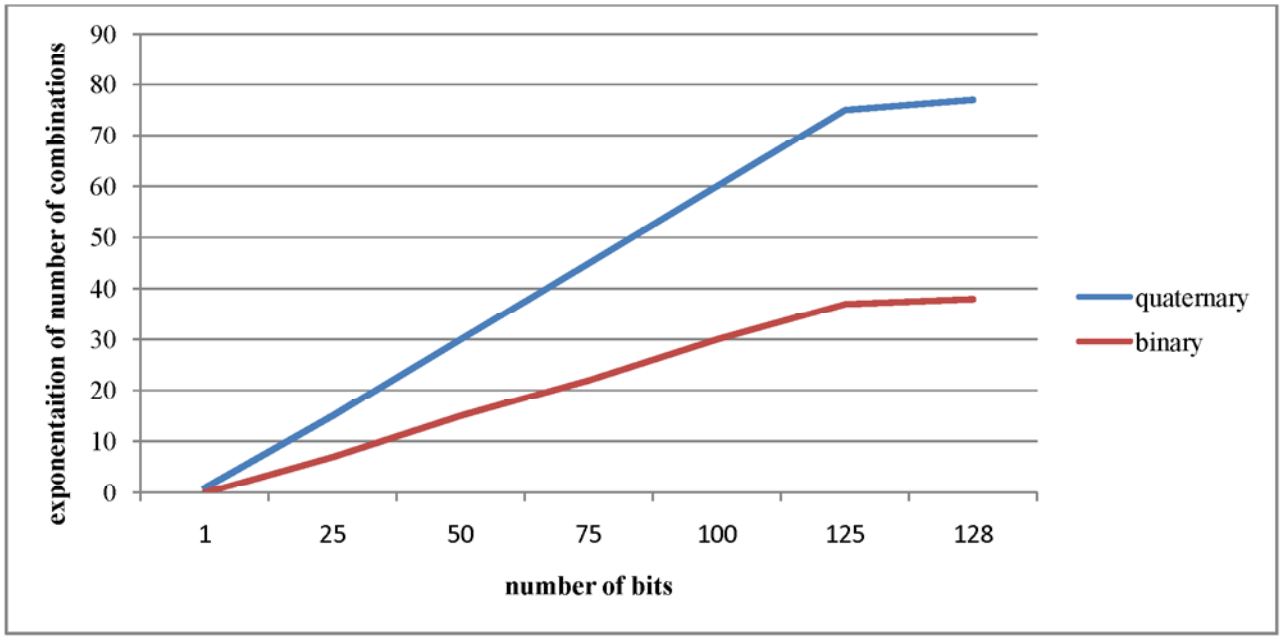


Figure 5.7: Shows number of bits versus number of combinations in terms of exponentiation

Table 5.3 : Number of combination versus number of combinations in quaternary and binary number systems

No of bits	1	25	50	75	100	125	128
quaternary	20	6E+15	6.76E+30	7.61E+45	8.57E+60	9.65E+75	9.88E+78
binary	2	67108862	2.25E+15	7.56E+22	2.54E+30	8.51E+37	6.8E+38

The data is embedded into the garbage file at random locations. A table *Tab* is also maintained to store the information of the address of the node in the file. The location is calculated as number of words from starting of the file (length of each word is of 3 characters each). The attributes of table *Tab* are <name of the node N , address addr of the node, length L of the data in the node> when encrypted using the method mentioned in section 5.3.1.

The address $addr_N$ is assigned randomly such that

addresss $addr_N$ ={ addr| addr < $addr_i$

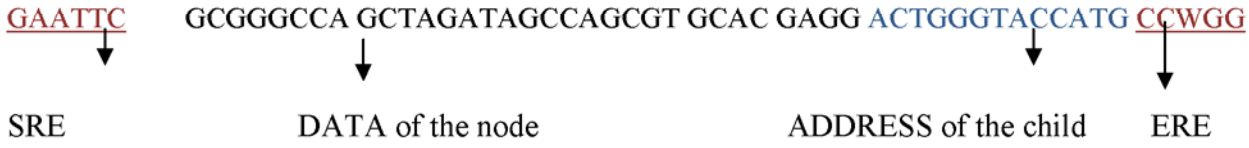
addr> $addr_i + l_i$}

where 1<=i<=n-1

l_i is the string length of the data at node i

The node is stored in the file and the address of the node is appended beside data of the parent. So that while traversing if one knows the node he can find all the children of the node in the garbage file. The location is also encrypted in the format as mentioned in section 5.3.1.

Now the strands will have a representation as shown in figure 5.8, stating that the whole subtree T starting from the node is visible to the client X.



If T2 is the subset of T_1 and it has to be sent to consumer y then head of sub tree T_2 will be prefixed and suffixed with <SRE_y, EREy >.Thus signifying that consumer y will only able to locate the head of T_2 and its children, but will not be able to search the address of the of its parent node in the file.

SRE	Data in the encrypted form	Address of leftmost child	……	Address of rightmost child	ERE

Figure 5.8: Diagrammatic representation of the node

The proposed secure dissemination technique is based on publish/subscribe. Whenever a consumer subscribes for the data, then the required file F' is sent to him.

The consumer will first locate the data starting with SRE and ERE. Then he will try to extract the data and child's addresses. And unfold rest of the tree till leaf is encountered.

5.4 Proposed Algorithm of the Secure Dissemination

The working of the algorithm for secure dissemination which implements the technique explained in section 5.3 involves the following two tasks:

- Execution at the server end as producer of the file
- Execution at consumer end to construct the sub tree

5.4.1 The Algorithm followed at Server End

i. Initialization:

a. Generate a garbage file F =∑{A,T,C,G} is prepared

ii. XML FILE is parsed and each tag is traversed using preorder traversal

a. r= preorder(T) /*For each node r
An entry is maintained in the table T */

A_N = randomaddress( ) /* randomly assign an address to each node to be inserted in the garbage file F in the garbage file. Such that it doesn't match with any earlier assigned address or any earlier assigned address+ length of the data.

Add (< T, Name of the node r, Address assigned to the node A_N, Length of the data of the traversed node>*/

iii. D= r-> info // D is the info at node

iv. $E_x (D_N)$ = encrypt(D) // Perform DNA encryption

v. List = ADD($E_x (D_N)$// maintain a list of data elements and its encrypted value

vi. GetSubscribedConsumer( )//For each client

a. LIST 1= add(SRE, ERE, n) in the table Enzymes // SRE and ERE to each consumer x such that the recognition site doesn't exits in the data. n number of bits of SRE and ERE.

vii. X'= prepXMLTemp() //Parse the XML file and according to access policy prefixing and postfixing for the XML tags which the consumer is authorized.

viii. F' = Modify(X') // The server then parses temporary XML file X' checks out if an tag is prefixed with for then SRE is picked from the table LIST1 and prefixes to $E_x (D_N)$. The data now transforms into $SRE_x E (D_N)$, where X is the consumer Number. Then the address location of the children are suffixed after looking into the table T.

The data now becomes $SREx E (D_N) A_{\text{leftmostchild of X}} \ldots A_{\text{rightmostchild of X.}}$

At the end the data is suffixed with ERE_x to show that the reading of the data has to end here. The data now becomes

$SRE_x E (D_N) A_{\text{leftmostchild of X,}} \ldots, A_{\text{rightmostchild of X}} ERE_x.$

The data is now embedded into the location which was entered in table T

ix. Transmit the prepared file F' to all the consumers

5.4.2 The algorithm followed at Consumer End

i. loc = Search(F', SRE) //Scan the SRE allocated to the consumer by the Server in the received garbage file F'. The position of last bit of SRE in F' is called as loc

ii. loc1 = Search(F', SRE) //Scan the ERE allocated to the consumer by the Server in the received garbage file F'. The position of last bit of ERE is called as loc1

iii. string1= readcontent(loc, loc1,F') //Read all the characters between loc and loc1 in string1. String1 contains $E (D_N) A_{\text{leftmostchild of N}} \ldots A_{\text{rightmostchild of N}}$

iv. $E (D_N)$ = Extractdata (string1) // Extract encrypted data from string1. Where N is the root of the subtree which the consumer is authorized to access.

v. Data= Decrypt(E (D_N)) // Decrypt the encoded data

vi. T = Root(data) // insert data as root node in subtree

vii. Repeat till all leafnode is reached

A = Extractaddresses(string1) // For each address ($A_{\text{leftmostchild of X}}$ .... $A_{\text{rightmostchild of X}}$)

string1= readcontent(loc, loc1,F')

E (D_X) = Extractdata (string1) //Where X is the node extracted

Data= Decrypt(E (D_X))

Insert(T, data, X)// insert data as a child under the node X

viii. Return (T)

5.5 Results and Discussion

The secure dissemination technique and the algorithm is discussed based on two point of views

- Probability of getting the right SRE and ERE
- Time taken to find the right SRE and ERE
- Requirement Satisfaction

5.5.1 Probability of Getting the Right SRE and ERE

If we consider the classical DNA encryption then there is a dataset of 4000 restriction enzyme. The number of options is $\binom{4000}{1} = 4000$. We can have at most 4000* 4000 number of combinations for choosing both SRE and ERE from a dataset of restriction enzymes. Hence the probability of getting the right set of enzyme is

$$p(A) = 1/(4000)^2$$

This limits the power of DNA encryption as it's easy to find the right set of restriction enzymes. To increase the number of combinations, we are not constraining the choice of restriction enzymes to classical dataset of restriction enzymes. We are customizing an enzyme in the following fashion to lower the probability to guess the right set of enzyme. The enzyme can be any combination of A, C, T, G which can span up to any length.

In the light of the above definition let us compute the probability of guessing the right set of enzyme:

The total number of combination is as follows

$$4 + 4^2 + 4^3 + 4^2 + 4^5 + \cdots + 4^n$$

So if the length enzyme is taken up to n length then the probability is

$$p(A) = {}^{1}/_{(4 + 4^2 + 4^3 + 4^4 + 4^5 + \cdots + 4^n)}$$

Hence the probability becomes $p(A) = \frac{3}{4(4^{n+1}-1)}$

Taking n=5 we get $p(A) = \frac{3}{4(4^6-1)} = 0.0001831.$

The figure 5.9 shows the probability of getting the right combination with respect to number of bits. It can be seen that the expression is highly convergent to zero.

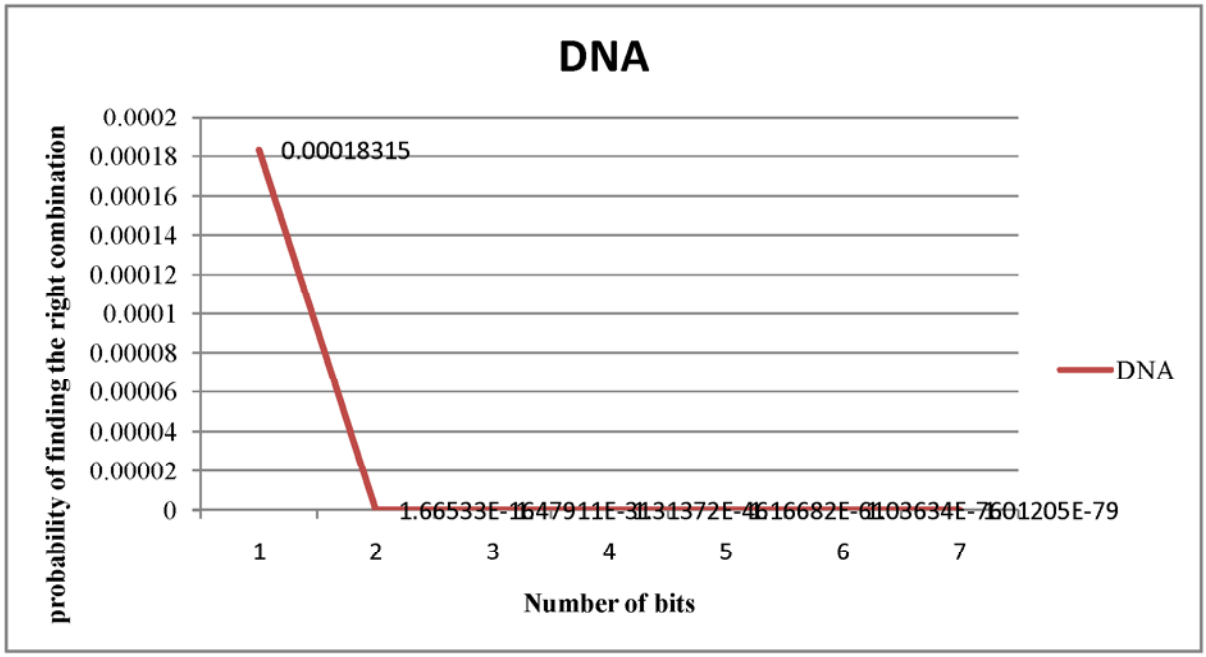


Figure 5.9: Graph to represents the number of bits versus the probility to find the right combination

5.5.2 Time Taken to Find the Right SRE and ERE

Time taken to find the right SRE or ERE depends on checking all possible key combinations until the correct key is found. This is also called as Brute-force attack[65-68] where systematically all possible key combinations are checked [69]. The calculation of the time taken is as follows:

Faster supercomputer: 10.51 Pentaflops = 10.51 E+15 Flops (Floating point operations per second)

Assume that Number of Flops required per combination check: 1000

No. of combination checks per second = (10.51 E+15) / 1000 = 10.51E+12

No. of seconds in one Year = 365 x 24 x 60 x 60 = 31536000

If $\propto$ is the number of combinations of a technique then number of years to crack = $\frac{\propto}{[(10.51\ E+12) \times 31536000]}$

Substituting the value of $\propto$ for different symmetric cryptosystem is shown in table 5.4.

Table 5.4: Time required cracking the combination in various types of symmetric cryptosystem.

Key size	DES (56 binary bits) [70]	AES (128 binary bits)	AES (192 binary bits)	AES (256 binary bits) [71]	DNA 128 quaternary bits
Possible combinations ($\propto$)	7.2E+16	3.403E+38	6.2E+57	1.1E+77	1.15E+77 [72]
Time to crack(years)	399secs	1.02E+18	1.87E+37	3.31E+56	3.4E+56

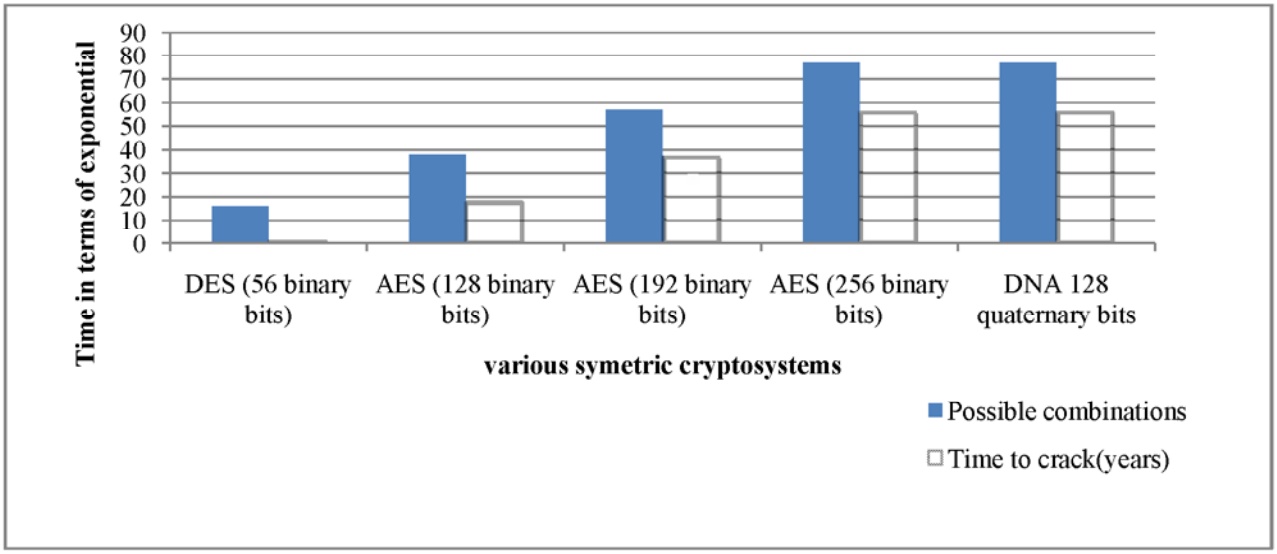


Figure 5.10: Graph for possible combination and year to crack the right combination with respect to the various symmetric cryptosystem.

It can be observed from the table 5.4 and figure 5.10 that the time taken to crack the right key / restriction enzyme is high in comparison to other symmetric cryptosystem. Thus proving that the technique is computationally secure [73].

5.5.3 Requirement satisfaction

The requirement of the problem was to ensure integrity, confidentiality and access control. Integrity is obtained as the probability of cracking the data is low as shown in the above section. Confidentiality and access control is also obtained as the consumer will be able to access data which he has subscribed to by the interface. The proposed technique support underlying nary tree structure of the XML document and hides the data in the form of linked list in a garbage file. The overhead is less as a single file is sent to all the consumers. The disadvantage of the solutions given by [29, 30] is if the Local XML structure changes, requires associated routing topology to be changed. Thus a subscriber needs to have a prior knowledge of the routing structure as the router structure as the router cannot fetch any content which is not hosted currently. We have implemented at server level, thus any change is just mentioned at the interface level. Thus reducing the cost and time required for a change in the system. This approach will not have scalability issues as other symmetric crypto systems. In [31, 36] the requestor asks for a set of

concepts and therefore requires knowledge of the ontological structure while in our work the ontological part is transparent for the requestor as the focus is on general access control.

Finally, our work is related to the secure XML broadcasting problem where the focus is the secure dissemination of XML documents to authorized users. In our approach we are specifying access policies using variable length key inspired from real world restriction enzymes, which is a computationally secure technique.

5.6 Summary

This chapter has proposed a secure dissemination technique to ensure that the consumers of the data are the legitimate ones according to the access policies. It presents a computationally secure technique in which there is a possibility to break the system theoretically but it's infeasible to do so by any known practical means. A multicast dissemination interface at the server/ producer end is proposed to implement the secure dissemination technique. Each client/consumer will subscribe in the interface and then automatically will be assigned a pair of randomly generated restriction enzymes called as SRE and ERE. The data will be appended with the SRE and ERE to signify that the data is meant for the respective consumer who has been assigned a particular SRE and ERE. Later the data is encrypted according to the technique and scattered in the garbage file. The garbage file is then transported to all the consumers where they will be able to view only the data as per the access control policies.

This chapter also highlights that due to the quaternary number system followed in the technique, it has very low probability of cracking the key. The number of years to crack the combination is also very high in comparison to various pre-existing symmetric cryptosystems like DES and AES. The results indicate that the proposed technique not only satisfies the requirement specification of secure dissemination but also points out its robustness in terms of time required to break the key. The time to crack the key is quite long and increases with increase in key length thus proving it to be computationally hard to crack by any known practical means.

The next chapter integrates the conclusion and contributions from chapter III, IV and V to reflect on the broad issues of interoperatability that formed the cornerstone of our research work.

CHAPTER 6: CONCLUSIONS AND CONTRIBUTIONS

The overall goal of our research work was to resolve the interoperatability issues in an enterprise web service platform. As the web services are under the heterogeneous ownership domains, there should be a uniform means to offer, discover and interact with each other. This chapter recapitulates the work that has been carried out as part of this research effort. It summarizes the conclusion that could be gained from the work on service selection & XML processing and security which form the basis of the research work. It provides an overview of the thesis structure. At the end it summarizes what has been learnt from this work and how these experiences contribute to the wider field of research.

6.1 Thesis Summary

Chapter I presented the concepts of SOA, how it can be achieved and what are its advantages and disadvantages. Chapter II presented the bulk of the contribution made by eminent researchers in the research domain of service selection, XML processing and security. Chapter III addressed the issue of service selection and the contribution of the optimized business service directory for the ESB web service platform. It showed how this will improve the integration of the web services which are under different administrative domains. Chapter IV addressed the issue of rewriting attacks and how it leads to insecure conversation. The chapter explained the three recommendations suggested to avoid and detect rewriting attacks and also proved that the overhead of incorporating these recommendations were significantly less. Chapter V presented the secure dissemination technique to avoid information leakage. A multicast dissemination interface at the server/ producer end is proposed to implement the secure dissemination technique. Incorporating all the three suggestion in the enterprise web service platform will resolve the interoperatabily issues.

6.2 Concluding Remarks

This chapter summarizes the results of the work described in chapter III, IV and V. The concluding remarks is gained from the two problems service selection and XML processing and security which form the basis of our research work.

6.2.1 Service Selection using Optimized Business Service Directory

One of the essential challenges in service selection is how to provide a suitable set of service candidates faster. Users demand systems that guarantee services with quality of Service (QoS) attributes like security, safety, flexibility of platforms reliability, performance, throughput and risk. SOA helps to provide value added services to the user with agility by following a publish-find-compose technique. The service selection is generally based on the QOS. The proposed Business service directory minimizes administrative overhead and increases usability by locating the best web service among the large number of functionally equivalent web services.

The results highlight that the time required to trigger 26 rules is less than the time required to triggers 243 rules. It was found that the time required by the rule generation technique using PSO and fuzzy clustering is faster than the alternative technique which uses fuzzy logic [4].

- The seek time of the webservice is less i.e., lesser are number of rules to trigger the less will be the seek time.
- The proposed optimized service registry will enable one to develop a better B2B or a B2C kind of e-commerce application with agility.
- The service requester can compare among the list of web service and choose the appropriate service provider based on its requirements.
- It automatically monitors the rank of the web service using the generated rules which makes it adaptive in nature.

6.2.2 SOAP Model for against rewriting attacks and insecure conversation

The rewriting attacks leads to insecure conversation as the contents of a SOAP message protected by an XML Signature as specified in WS-Security can be altered without invalidating the signature. The proposed SOAP model avoids rewriting attacks and ensures secure conversation. The model highlighted three possible recommendations namely, using shared key for encrypting timestamp in the message body for generating corresponding signature; Secondly, using value referencing both for signature validation and message processing; and finally encrypting the whole SOAP body instead of sending an open SOAP Message in the network to prevent unauthorized access.

The features of the model constitute of the following points

- The referencing to the node in this model is done using value hence the reference is called as *value referencing*. Thus it does not limit the abilities of defining signature reference.
- The model sticks to the basic structure of SOAP model mentioned in WS standards.
- We are not attaching the extra information, we are just attaching timestamp and embedding it into SOAP body and then encrypting the whole SOAP body. Thus the malicious attacker cannot access the added information

The comparison in section 4.3.1 showed that the proposed SOAP model is better than the earlier SOAP model. It also has less overhead in terms of performance metric time which is an important issue in security. It was observed that growth in time required for message encryption and decryption versus the number of tags in the SOAP message was .04% and 0.14%. It was also seen that the Time required to encrypting the signature with respect to number of tags in the SOAP message had a growth of 0.5%. Thus it can be inferred that the proposed SOAP model doesn't have much overhead.

6.2.3 Proposed Secure Dissemination Model

Secure dissemination of an XML file is one of the techniques to ensure data integrity and confidentiality. The research work presents a secure dissemination technique which ensures that extraneous is inaccessible even if the consumer is a legitimate consumer. Consequently, this avoids information leak. The technique applies DNA cryptography due to its feature of compactness and simplicity. The technique encrypts the data and hides it in a garbage file, such that only legitimate consumer can see only the subscribed amount of data according to the access policies. It also presents multicast dissemination interface that implements the proposed technique at the server level. The interface is built dynamically and asynchronously using publish–subscribe middleware which is able to perform selective XML content delivery.

The model has the following features:

- The model supports the structure of the data in the form of N-ary tree. The tree at the consumer end can be easily reconstructed.
- The time taken to crack the right key / restriction enzyme is high in comparison to other symmetric cryptosystem.
- It was seen that the probability of finding the right key is highly convergent to zero

6.3 Contributions

Here we summarize the main contributions and achievements of the research carried out as part of this thesis.

i. Reduced Seek Time

The time required to search the web service in the database also called as seek time will be significantly less as it will be dependent on the number of rules required to trigger and the number of rules is less in the proposed technique as compared to implementing the BSD using fuzzy logic. The time complexity to parse the rules is O (N) where N is the no of rules. Lesser the number of rules to trigger less will be the seek time. The seek time is also affected by the database size and the number of quality attributes considered qualifying the web service.

ii. Intelligent System

The rules are successfully generated automatically using dataset thus making the system intelligent, in comparison to the technique using fuzzy logic where the rules are to be entered by a human expert. Human intervention makes the system error prone and manual. The quality of rules is dependent on the training dataset; the data should be less overlapping and should have all varieties of output. The rules can still be generated using less number of entries in the dataset.

iii. Adaptive in nature

The rules are adaptive i.e., any change in the dataset or the ranking criteria will automatically be reflected in the rules and thus a new set of rules will be generated. The web services can be ranked according to the new rules.

iv. Early Detection of Rewriting Attacks

The proposed SOAP model helps the consumer to find detect that the SOAP message in conversation has suffered rewriting attack. If there is any change in the SOAP message, then the signatures will never match. Thus the signature invalidation proved that there is a rewriting attack.

v. Ensuring Secure Conversation

The proposed SOAP Model ensures secure conversation as a non-authorized person can ever tamper the message as it is encrypted. If a trusted entity in the conversation breaks the trust and sends the message to some other after modifying it and pretends that he is just a sender not the constructor. The message can be validated whether it was modified in the transit, by validating against the signature and the timestamp.

vi. Time Independent

One of the outcome of the research work highlights that the increase in the length of the SOAP message with tags is independent of the time required for encryption and decryption. A graph plot of time required to encrypt the message for sending with respect to number of tags in the SOAP message showed that the percentage of growth in the time with respect to the number of tags was 0.14%. Another graph plotted shows that the time required to encrypt the signature w.r.to the number of tags in the SOAP message shows that the percentage of growth is 0.5%.

vii. Computationally Secure

The proposed secure dissemination technique present a computationally secure technique . There is a possibility to break the system theoretically but it's infeasible to do so by any known practical means as the time to crack the key is quite long and increases with increase in key length thus proving it to be computationally hard to crack by any known practical means.

viii. Superior 4- ary System

The 4- ary number system considered makes more number of combinations than

various other existing symmetric cryptosystem as the number of combinations to crack the key is much larger.

ix. XML Encryption in a DNA Strand

Proposed secure dissemination technique secures the data in a DNA strand and provides data integrity. It ensures that the consumers of the data are the legitimate ones according to the access policies.

x. Multicast Dissemination Interface

A multicast dissemination interface at the server/ producer end is proposed to implement the secure dissemination technique. Each client/consumer will subscribe in the interface and then automatically will be assigned a pair of randomly generated restriction enzymes called as SRE and ERE. The data will be appended with the SRE and ERE to signify that the data is meant for the respective consumer who has been assigned a particular SRE and ERE. Later the data is encrypted according to the technique and scattered in the garbage file. The garbage file is then transported to all the consumers where they will be able to view only the data as per the access control policies.

6.4 Future Work

The future work will involve optimizing and integrating the elements of the ESB platform incrementally. We would like to extend the proposed service selection algorithm to locate the web service semantically as well. We also aim to use software agents for service selection. Agents can be allocated for selecting a web service in a zone. We plan to further investigate to find out how other access control policies and integrity models can be implemented using the proposed secure dissemination technique. We further plan to address other interoperatability issues like service level agreements, to enhance the performance of ESB web service platform.

6.5 Summary

At the end the research work attempts to bridge the gap in the SOA development life cycle primarily focusing on the issue of service selection and security. The three proposed artifacts namely BSD, SOAP model and secure dissemination technique fulfils the stated objectives of

better service selection and security. Incorporating these proposed changes would ensure better and adaptable service selection. It not only successfully detects rewriting attacks and establishes secure conversation in the to-and-fro message transmission but also satisfies the requirement specification of secure dissemination by pointing out its robustness by being computationally secure.

References

[1] J. A. Zachman, "A Framework for Information Systems Architecture", IBM Systems Journal, Vol. 26, Issue 3,pp. 276–292, 1987.

[2] L.Brien, P.Merson, L.Bass, "Quality Attributes for Service-Oriented Architectures", Proceedings of the International Workshop on Systems Development in SOA Environments, Minneapolis, MN, pp.3, 2007

[3] J. Leon Zhao, T. Mohan, L. Liliang, "Services Computing as the Foundation of Enterprise Agility: Overview of Recent Advances and Introduction to the Special Issue", Information systems Front, Vol. 9, Issue 1,pp.1-8, 2007

[4] L.-J. Zhang, J. Zhang, and H. Cai, in Services Computing, Springer and Tsinghua Univ. Press, July 2007

[5] M. Keen, S. Bishop, Patterns: Implementing an SOA Using an Enterprise Service Bus, IBM Red book, July 2004

[6] E.Newcomer, Understanding SOA with Web Services, Pearson Education India, 2005

[7] Q.Yu, X.Liu, A. Bouguettaya , B. Medjahed, "Deploying and Managing Webservices: Issues, Solutions, and Directions", The VLDB Journal — The International Journal on Very Large Data Bases, Vol. 17 ,Issue 3, pp. 537- 572 , May 2008

[8] Lewis, Grace, Smith, Dennis, Proceedings of the International Workshop on the Foundations of Service-Oriented Architecture (FSOA 2007) (CMU/SEI-2008-SR-011). Software Engineering Institute, Carnegie Mellon University, 2008.

[9] D.E.Cox,"Management of the Service-Oriented-Architecture Life Cycle", IBM Systems Journal Vol. 44, Issue 3, pp.709 - 726 , 2005

[10] Shuangxi Huang, Yushun Fan," Model Driven and Service Oriented Enterprise Integration---The Method", Proceedings of the Framework and Platform Sixth International Conference on Advanced Language Processing and Web Information Technology, Henan, China, pp. 504 - 509, 2007

[11] A. Segev, E. Toch, "Context-Based Matching and Ranking of Web Services for Composition", IEEE Transactions on Services Computing, Vol .2, Issue. 3 pp. 210 – 222, 2009

[12]K. Kritikos, D.Plexousakis, "Requirements for QoS-based Web Service Description and Discovery", IEEE Transactions on Service Computing, Vol. 2,Issue. 4, pp. 321-336 ,2009

[13] T. Pilioura, Tsalgatidou, A. "Unified publication and Discovery of Semantic Web services" ACM Transaction on Web, Vol.3, Issue.3, Article 11, 44 pages, 2009

[14] Y.Baocai,Y. Huirong,F.Pengbin, G. Liheng, L.Mingli, "A framework and QoS based web services discovery" Proceedings of the IEEE International Conference Software Engineering and Service Sciences (ICSESS), Beijing, China, pp 755 – 758, 2010

[15] T. Yu, Y. Zhang, K. Lin, "Efficient algorithms for Web services selection with end-to-end QoS constraints", ACM Transaction on Web, Vol.1, Issue.1, Article 6, 26 pages, 2007.

[16] V. Xuan Tran, H. Tsuji, "QoS based Ranking for Web Services: Fuzzy Approaches", Proceedings of the 4th International Conference on Next Generation Web Services Practices, Seoul, South Korea, pp. 77 – 82, 2008

[17] M.Tang , L. Ai ,"A Hybrid Genetic Algorithm for the Optimal Constrained Web Service Selection Problem in Web Service Composition" ,Proceedings Of IEEE Congress on the Evolutionary Computation (CEC), Barcelona, Spain, pp. 1 – 8 , 2010

[18] E. Al-Masri, Q.H.Mahmoud," Discovering the Best Web Service: A Neural Network-based Solution" ,Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, 2009, San Antonio, Texas, pp. 4250 – 4255, 2009

[19]R. Mohana,D. Dahiya," Approach and Impact of a Protocol for Selection of Service in Web Service Platform" , ACM SIGSOFT Software Engineering Notes ,Vol. 37 Issue 1, pp. 1-6 . 2012

[20] M. McIntosh and P. Austel, "XML Signature Element Wrapping Attacks and Countermeasures ",Proceedings Of the workshop on Secure web services , VA, USA , Pp. 20 – 27, 2005

[21] S.Gajek, M.Jensen, L. Liao, J.orgSchwenk ,"Analysis of Signature Wrapping Attacks and Countermeasures", Proceedings of IEEE International Conference on Web services, ICWS, CA, USA ,pp. 575 – 582, 2009

[22] A.Benameur,"XML Rewriting Attacks: Existing Solutions and Their Limitations", Proceedings of the ACM workshop on Secure Web Services , New York, NY, USA, pp. 53-60, 2008

[23] M. AshiqurRahaman, A. Schaad, " SOAP-based Secure Conversation and Collaboration", Proceedings of IEEE International Conference on Web services, Salt Lake City, USA , pp. 471 – 480, 2007

[24] Y. Liu, H. Zhao, Y.i Li," Research on Secure Transmission of messages", Proceedings of 12th International Conference on Computer Supported Cooperative Work in Design, Xi'an, China, pp. 771 – 776, 2008.

[25] S. K. Sinha, A.Benameur,"A Formal Solution to Rewriting Attacks on SOAP Messages", Proceedings of ACM Workshop on Secure Web Services SWS '08 , Virginia, USA, pp. 53 – 60, 2008 .

[26] N.Sidharth, J.Liu ,"Intrusion Resistant SOAP Messaging with IAPF", Proceedings of IEEE Asia- Pacific Services Computing Conference, Yilan, Taiwan, pp. 856-862 , 2008

[27]N.Sidharth, J.Liu, "A Framework for Enhancing Web services Security," Proceedings of 31st annual International Computer Software and Applications Conference , Beijing, China, pp. 23-30, 2007

[28] M. Srivatsa and L. Liu, "Securing Publish–Subscribe Overlay Services with Eventguard", Proceedings of 12th ACM Conference on Computing Communication and Security, pp. 289–298, 2005.

[29] E. Bertino, E.Ferrari,"Secure and Selective Dissemination of XML documents" ,ACM Transactions on Information and System Security - TISSEC , Vol. 5, Issue.3, pp. 290-331, 2002

[30] A. Kundu, E.Bertino," A New Model for Secure Dissemination of XML Content", IEEE Transaction on Systems, Man, and Cybernetics—Part C: Applications and Reviews, Vol. 38, Issue. 3, pp.292-301, MAY 2008

[31]M. A. Rahaman, Y. Roudier, P. Miseldine, and A. Schaad," Ontology-based Secure XML Content Distribution", Proceedings of 24th International Information Security Conference, Pafos, Cyprus, pp. 294-306 May 2009.

[32] W.-C. L. Bo Luo, Dongwon Lee and P. Liu,"A Flexible Framework for Architecting XML Access Control Enforcement Mechanisms",Vol. 3178/2004 of Lecture Notes in Computer Science. Springer Berlin/ Heidelberg, December 2004.

[33] E. Damiani, S. D. C. di Vimercati, S. Paraboschi, and P. Samarati," Fine Grained Access Control for SOAP Eservices", In Proceedings of the 10th International Conference on World Wide Web, New York, NY, USA , pp. 504-513, 2001

[34] W. Fan, C.-Y. Chan, and M. Garofalakis," Secure XML Querying With Security Views", In Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data", pp. 587-598, New York, USA, 2004.

[35] G. Kuper, F. Massacci, and N. Rassadko, "Generalized XML Security Views", In Proceedings of the Tenth ACM symposium on Access Control Models and Technologies, pp. 77-84, New York, NY, USA,2005.

[36] M. Murata, A. Tozawa, M. Kudo, and S. Hada," XML Access Control Using Static Analysis", Proceedings of the 10th ACM conference on Computer and Communications Security, pp. 73-84, New York, USA, 2003.

[37] L.A Zadeh, "Fuzzy Sets", Information and Control, Vol. 8, Issue 3, pp. 338-353, 1965

[38] L.A. Zadeh **,"**The Role of Fuzzy Logic in the Management of Uncertainty in Expert Systems **",**Fuzzy Sets and Systems, Vol. 11, Issues 1, pp. 197–198, 1983

[39] http://www.austinlinks.com/Fuzzy/expert-systems.html as on July, 2011

[40] http://en.wikipedia.org/wiki/Cluster_analysis on July, 2011

[41] D.tikk et al.,"A Survey on Universal Approximation and its Limits in Soft Computing Techniques", International Journal of Approximate Reasoning, Vol. 33, Issue 2, pp. 185–202, 2003.

[42] J. Kennedy, "The Particle Swarm: Social adaptation of knowledge", Proceedings of the International Conference on Evolutionary Computation, Alaska, USA, pp. 303-308, 1997

[43] L.B Zhang, "Solving Multi Objective Optimization Problems Using Particle Swarm Optimization", in the Proceedings of the IEEE on Evolutionary Computation, pp.2400 – 2405, USA 2003.

[44] http://www.uoguelph.ca/~qmahmoud/qws/index.html as on Jan 2011

[45] E. Al-Masri, Q. H. Mahmoud, "Discovering the best webservice", Proceedings of 16th International Conference on World Wide Web (WWW), Alberta, Canada (poster), pp. 1257-1258, 2007

[46] E. Al-Masri, Q. H. Mahmoud, "QoS-based Discovery and Ranking of Webservices", Proceedings of IEEE 16th International Conference on Computer Communications and Networks (ICCCN), Hawaii USA, Pp. 529-534,2007

[47] http://www.w3schools.com/soap/default.asp as on Jan, 2013

[48] A. Berglund, S. Boag, D. Chamberlin, M. F. Fernández, et al. (2007).,"XML path language (XPath) 2.0."W3C Recommendation.

[49] N. Gruschka, M. Jensen, F. Kohlar and L.Liao "On Interoperability Failures in WS-Security: The XML Signature Wrapping Attack" in Electronic Business Interoperability: Concepts, Opportunities and Challenges ,IGI Global , pp. 615-635, March, 2011.

[50] http://www.w3.org/TR/xpath as on July, 2012

[51] F. Wang, "A space efficient XML DOM parser" , Data & Knowledge Engineering, Vol. 60, Issue 1, Pp: 185–207, 2007

[52] D. E. Knuth Art of Computer Programming, Volume 1: Fundamental Algorithms (3rd Edition)

[53] Xing Wang, Qiang Zhang, " DNA Computing-Based Cryptography", Fourth International Conference on Bio-Inspired Computing, 2009, pp: 1 – 3.

[54] A. Leier, "Cryptography with DNA Binary Strands", Biosystems, Vol.57, Issue 1, pp. 13–22 , June 2000

[55] A. Gehani, T. LaBean, and J. Reif, "DNA-Based Cryptography," Aspects of Molecular Computing Lecture Notes in Computer Science, Vol 2950, pp: 167-188, 2004

[56]S. V. Kartalopoulos, "DNA-Inspired Cryptographic Method in Optical Communications, Authentication and Data Mimicking" in the Proceedings of the IEEE on Military Communications Conference, Atlantic City, NJ, Vol.2, Pp:772-779, 2005

[57]K. Tanaka, A. Okamoto and I. Saito, "Public-key System using DNA as a One-Way Function for Key Distribution", Bio systems, Vol. 81, Issue 1, pp:25-29, 2005

[58] A. Gehani, T. LaBean and J. Reif, "DNA-based cryptography", Lecture Notes in Computer Science, Vol.2950, pp.167-188, 2002

[59] M.SAEB , E. EL-ABD , M. E. EL-ZANATY "On Covert Data Communication Channels Employing DNA Recombinant and Mutagenesis-based Steganographic Techniques" in the Proceedings of International Conference on Computer Engineering and Applications (WSEAS), Wisconsin, USA, pp. 200-206 , 2007

[60] http://rebase.neb.com/rebase/rebase.enz.html as on August, 2012

[61] http://www.neb.com/nebecomm/products/category1.asp as on August, 2012

[62] X. Wang, Q. Zhang" DNA Computing-based Cryptography",Fourth International Conference on Bio-Inspired Computing, pp. 1 – 3, 2009

[63] A. Delgado, "DNA chips as Lookup Tables for Rule-Based Systems", Engineering Science and Education Journal Vol. 11, Issue .3 pp .99 - 105, Jun 2002

[64] G. Cui , L. Qin, Y. Wang , X. Zhang, "An Encryption Scheme Using DNA Technology", 3rd International Conference on Bio-Inspired Computing: Theories and Applications, pp. 37 – 42, 2008.

[65] J.-Sik Cho ,"Securing Against Brute-Force Attack: A Hash-based RFID Mutual Authentication Protocol Using a Secret Value" Computer Communications, Vol.34, Issue .3, 1 pp.391–397, 2011

[66] R. E. Korf, "Planning As Search: A Quantitative Approach", Artificial Intelligence Vol. 33, Issue 1, pp. 65–88, 1987

[67] A. J. McCoy, R. W. Grosse-Kunstleve, L. C. Storoni and R. J. Read, "Likelihood-Enhanced Fast Translation Functions", Acta Crystallographica Biological Crystallography Vol. 61, pp. 458-464 ,2005

[68] M. J. Wiener, "The Full Cost of Cryptanalytic Attacks", Journal of Cryptology, pp. 105–124, 2004

[69]http://www.eetimes.com/design/embedded-internet-design/4372428/How-secure-is-AES-against-brute-force-attacks- as on August, 2012

[70] C.-Chung Lu, "Integrated Design of AES (Advanced Encryption Standard) Encrypter and Decrypter", Proceedings of the IEEE International Conference on Application-Specific Systems, Architectures, and Processors, pp. 277 – 285, 2009

[71] D. Salama A. Elminaam," Evaluating the Performance of Symmetric Encryption Algorithms", International Journal of Network Security, Vol.10, Issue.3, pp.216–222, 2010

[72]http://www.plosone.org/article/info%3Adoi%2F10.1371%2Fjournal.pone.0044212#pone.0044212-Lee1 as on August, 2012

[73] M. Naor," Computationally Secure Oblivious Transfer", Journal of cryptology, Vol.10, Issue.3, pp. 1-5

LIST OF PUBLICATIONS

Journals

1. Rajni Mohana and Deepak Dahiya. *"Specifying Access Policies for Secure Content Dissemination of XML: A Technique Inspired by DNA Cryptography"*. Paper published in the Journal of Computing and Information Technology (CIT), Vol. 21, No. 2, University of Zagreb, Croatia (ISSN 1330-1136), pp. 1 - 14.
2. Rajni Mohana and Dahiya. *"Addressing Interoperability Failures in WS-Security:A SOAP Model Against Rewriting Attacks and Insecure Conversation"*, Paper accepted for publication in the International Journal of Computers and Their Applications (IJCA), US (Accepted for Publication).
3. Rajni Mohana and Deepak Dahiya. *"An Optimized Business Service Directory for the ESBPlatform in SOA"*. Paper published in the International Journal of Computer Networks and Communications (IJCNC), Vol. 4, No. 5, September 2012, India (ISSN 0975 – 2293), pp. 165- 186.
4. Rajni Mohana and Deepak Dahiya. "*Approach and Impact of a Protocol for Selection of Service in Web Service Platform* " Paper published in the ACM SIGSOFT Software Engineering Notes(SEN), USA, Vol. 37, No. 1, January 2012, pp. 1 - 6.
5. Rajni Mohana and Deepak Dahiya. "A Proposed SOAP Model Against Wrapping Attacks and Insecure Conversation". Paper published in the International Journal of Computer Science Issues (IJCSI), Vol. 10, Issue 2, No 3, March 2013, (ISSN: 1694-0814), pp. 151 - 156.

Conferences

1. Rajni Mohana and Deepak Dahiya. *"Optimized Service Discovery using QoS basedRanking: A Fuzzy Clustering and Particle Swarm Optimization Approach,"* Paper published in the IEEE Proceedings of the 35th IEEE Computer Software and Applications Conference, Munich, Germany (IEEE COMPSAC 2011), pp. 452 – 457.
2. Rajni Mohana and Deepak Dahiya. "*Designing QoS based Service discovery as a Fuzzy Expert System*". Paper published in the Springer Series in Communications in Computer and Information Science (CCIS) of the 4th International Conference on Contemporary Computing, Jaypee Institute of Information Technology, Noida, India, (IC3 2011), pp. 533 - 534. (Poster)

3. Atul Saurabh, Deepak Dahiya and Rajni Mohana. "*Maximizing Automatic Code Generation: Using XML Based MDA*". Paper published in the Springer Series in Communications in Computer and Information Science (CCIS) of the 5th International Conference on Contemporary Computing, Jaypee Institute of Information Technology, Noida, India, (IC3 2012), pp. 283 - 293.