# DEVELOPMENT OF FRAMEWORK FOR IMPROVING SECURITY MECHANISMS IN DESIGNING DISTRIBUTED SYSTEM

**THESIS SUBMITTED FOR PARTIAL FULFILLMENT OF THE AWARD OF DEGREE OF**

**Doctor of Philosophy**
***in***
**Computer Science & Engineering**

*By*
**Vijay Prakash**

*Under the Supervision of*
**Dr. Manuj Darbari**

**Department of Computer Science & Engineering**
**School of Engineering**
**Babu Banarasi Das University**
**Lucknow 226 028 (U.P.), India**

**April, 2015**

## Certificate of the Supervisor

This is to certify that the thesis entitled **“Development of Framework for Improving Security Mechanisms in Designing Distributed System”** submitted by **Mr. Vijay Prakash** for the award of Degree of Doctor of Philosophy in Computer Science and Engineering from Babu Banarasi Das University, Lucknow (Uttar Pradesh), is a record of authenticate work carried out by him under my supervision. To the best of my knowledge, the matter embodied in this thesis is the original work of the candidate and has not been submitted elsewhere for the award of any other degree or diploma.

**Dr. Manuj Darbari**
Professor
Department of Computer Science & Engineering
Babu Banarasi Das University,
Lucknow

# Declaration

I, hereby, declare that the work presented in this thesis entitled “Development of Framework for Improving Security Mechanisms in Designing Distributed System” in fulfillment of the requirements for the Degree of Doctor of Philosophy, Department of Computer Science and Engineering, Babu Banarasi Das University, Lucknow is an authentic record of my own research work carried out under the supervision of Dr. Manuj Darbari.

I also declare the work embodied in the present thesis is my original work and has not been submitted by me for any other degree or diploma of any university or institution.

**(Vijay Prakash)**

# CONTENTS

# ACKNOWLEDGEMENTS

I take the opportunity to express my profound gratitude and deep regards to my guide Dr. Manuj Darbari, Professor, Department of Computer Science & Engineering, Babu Banarasi Das University, Lucknow for his exemplary guidance, monitoring and constant encouragement throughout the course of this thesis. The blessings, help and guidance given by him time to time shall carry me a long way in the journey of life on which I am about to embark.

I would also like to thank Dr. A. K. Mittal Vice Chancellor, Babu Banarasi Das University, Dr. Seethalekshmi K., Dean, School of Engineering, Babu Banarasi Das University, Dr. S. Ahmad Ali, Dean, Applied Sciences, Babu Banarasi Das University and Dr. V. K. Singh, Head, Department of Information Technology, Babu Banarasi Das National Institute of Technology & Management for their constant support, encouragement under this aura of academic excalibur, their enormous experience, their sincerity towards their duty which have truly inspired me leaps and bounds.

I must not forget to appreciate the extreme patience and support of my wife Mrs. Parul Srivastava during this work. I am really thankful to her. I also owe this research work to my loving daughters Ratika and Nitya who were extremely affectionate throughout.

I thank my parent, sister, brothers and friends for their constant encouragement without which this assignment would not be possible.

Last but not the least I would like to thank the almighty for showering on me his choicest blessing and helping me to achieve my goal.

**- VIJAY PRAKASH**

# PREFACE

A distributed system can be defined as a group of independent computers that appear to the users of the system as a single system. Some of the examples of the distributed systems are network of workstations, automated assembly lines, a network of small office computers. Each computer executes components and works on a distributed middle-ware this distributed middle-ware enables the components to coordinate their activities.

Application requirements can be functional requirements depending on the centralized system and distributed systems. In non functional requirements various parameter are scalability, concurrency, openness, and resource sharing and fault tolerance. The summarized definitions of these requirements are as follows - Scalability this denotes the ability to accommodate a growing load in the future; Concurrency deals with the execution of different events and conflicting issues; Openness the authorization requirements are totally dependent on the distributed system requirements; Resource sharing offer resources i.e. hardware, software and data required to be shared by more than one user. Distributed objects provide a sophisticated model of Resource sharing. Fault tolerance means that the operation can continue even in the presence of fault and it is achieved in the distributed system by means of replication.

A cryptosystem is defined as the system which transforms plaintext into cipher text using a key (public or private). The process of designing a cryptosystem is called cryptology whereas the study of cryptosystem is called cryptanalysis. The fusion of cryptology and cryptanalysis is called cryptography. The cryptography is composed of two theories, namely

Symmetric key cryptography and Public key cryptography. This provides security by massage integrity and digital signatures. Several features of the cryptography are as follows

*Confidentiality:* In this phase it is ensured that only authenticated sender and intended receiver should understand the original message. In this procedure sender encrypts message and receiver decrypt the message.

End point authentication: In this phase the sender and receiver are confirming the identities of each other.

*Message Integrity:* In this phase the sender and receiver would be ensuring that the original message was not altered during inframsist and after worlds.

Preserving security and privacy is a challenging issue in distributed systems. This proposal makes a step forward in solving this issue by proposing a generic framework for multi-factor authentication to protect services and resources from unauthorized use. The authentication would be based on trust based model. Proposed framework would not only demonstrate how to obtain secure multi-factor authentication, but also would be addressing several prominent issues of biometric authentication in distributed systems (e.g., client privacy and error tolerance). The model would also be satisfying the Trust Based Model properties.

The objectives of the proposed research work are to enhance the security of the distributed system. This proposes the new model of security framework based on the DNA cryptography and computational intelligence approach. The major findings of the proposed research work are

- Computational intelligence approach is used to design a trust based system security framework. Fuzzy logic is used to approximate the trust values of entities in the proposed system.
- DNA cryptography is used to enhance the robustness of designed system. DNA concept is used for the decryption and encryption.

# CHAPTER 1

# INTRODUCTION

## 1.1 OVERVIEW

The recent developments in security in Distributed Systems propose to extend notions developed for stand-alone system security in the development of requirements for network (distributed system) security as discussed by Firdhous (2011). This work advocates the use of network topology in concert with the evaluation levels of the systems in the network (*i.e.*, its nodes) to determine constraints that should be imposed on classified processing. In a single proposal, several modes of network operation are defined and constraints on network topology and system evaluation levels are developed for each mode.

The appeal of these proposals is conceptual parsimony (Xu *et al.* (2003)). If it can be shown that existing computer security concept are adequate to describe security in a distributed environment, a great deal of expense, intellectual effort, and institutional retooling can be eliminated. This is a worthy goal deserving serious attention.

However, there is danger in this approach. If the expected advantages are great, much effort will be made attempting to provide the necessary generalizing framework, while little attention will be paid to the underlying question of whether this generalization is possible as discussed by Yao and Fidelis (2003). The temptation is to assume the possibility without clearly establishing the merits of that assumption.

This proposal presents factors other than node evaluation levels and network topology that affect distributed system security. Its scope is limited to establishing these additional factors as relevant.

## 1.2 BACKGROUND

A distributed system (Aikebear *et al.* (2011)) can be defined as a group of independent computers that appear to the users of the system as a single system, some of the examples of the distributed systems are: network of workstations, automated assembly lines, a network of small office computers. Each computer executes components and works on a distributed middle-ware this distributed middle-ware enables the components to coordinate their activities.

The various requirements of distributed systems are categorized as application requirements and non-functional requirements. The application requirements can be functional requirements depending on the centralized system and distributed systems. In non functional requirements various parameter are scalability, concurrency, openness, and resource sharing and fault tolerance as discussed by Firdhous (2011). The summarized definitions of these requirements are as follows.

*Scalability* defines the ability to accommodate a growing load in the future.

*Concurrency* deals with the execution of different events and conflicting issues.

*Openness* the authorization requirements are totally dependent on the distributed system requirements.

*Resource sharing* offers resources i.e. hardware, software and data required to be shared by more than one user.

Distributed objects provide a sophisticated model of Resource sharing. Fault tolerance means that the operation can continue even in the presence of fault and it is achieved in the distributed system by means of replication.

### 1.2.1 Design Issues of Distributed System

The design issues for the distributed systems have been discussed by Pallickara *et al.* (2007) and Shehab *et al.* (2010) which for our purpose are recorded below.

1. *Transparency*. This directly relates to achieve the single system concept means how a group of computer can be presented as a single computer. It can be achieved by using two levels. In level 1, the distribution from users is hidden while in the second level the transparency to the program is maintained. The number of transparency includes:

   (i) *Location Transparency.* In this user's doesn't know about the place of hardware and software recourse such as CPU, Printers, files and databases.

   (ii) *Migration Transparency.* In this transparency it is assured that the resources are free to change their location without changing their names.

   (iii) *Replication Transparency.* In this the files and resources can be replicated.

(iv) *Concurrency transparency*. In this transparency, the multiple users are allowed to concurrently access the same resource using lock, unlock and mutual exclusion.

2. *Flexibility***.** Flexibility makes the system easy to change.
3. *Reliability.* The distributed systems must be more reliable than the single system they need to maintain consistency, security and fault tolerance capabilities.
4. *Performance.* The Performance of the distributed system is affected by communication delays and lack of fault tolerance.
5. *Scalability.* System grows with time depending on the requirement of resources. When the requirement of resource increases linearly in terms of the size of the system, the distributed systems are not scalable.

The distributed system security as discussed by Firdhous (2011) is an important research issue. There are many security aspects that can be incorporated to make distributed system secure. Some of them are briefly discussed below.

1. *Security for Computing Clusters.* When the computing clusters as discussed by Xie and Qin (2008) are made available to the public using public resources such as internet, various attacks can be attempted during inter node communication and cluster service communication.
2. *Grid Systems Security.* Many Security mechanics have been developed to secure the grid resources as discussed by Demchenko *et al.* (2008) against several attacks. Middle-ware while is a critical system software in a grid infrastructure gives the common communication infrastructure. The modules for security purpose are grid certification authority certificate and grid authentication modules which is the critical components in preventing external users.

3. *Distributed Database Seemly.* The Implementation security in distributed database is a critical research issue dealing with new types of databases like object oriented database, temporal database, object relational database. A multilevel secure database management system (MLS/DBMS) has been developed for security database operations as given in Kaur *et al.* (2005). Zubi (2009) has proposed a multilevel access control confidentially, reliability, recovery to maintain the distributed database system security.

4. *Distributed Storage System Security.* This model addresses the security aspect in the distributed storage of data. The various addressed security issues are confidentiality, integrity, availability and authentication as give in Kaulman *et al.* (2002).

### 1.2.2 Trust Based Security Issues in Distributed System

There are various security issues in distributed system such as Authentication, Authorization, Confidentially, Data Integrity and Denial of Service Attack. The concepts of trust Li and Singhal (2007) in ensuring the distributed system security have been frequently used in recent past. Generally, an entity can be called to trust a second entity when the first entity makes the assumptions that the second entity will behave exactly as the first entity expects following outlines are considered for modeling trust in security architecture (I) Trust is a key attribute of security architecture; (II) Security architecture should ensure a level of trust; (III) Trust is enhancing the confidence that something will or will not happen in a predictable and defined manner.

This is a coherent framework for studying the security polices credentials and relationships of trust among different entities as discussed by Wang and Sun (2009). There are two types of models for trust management, *viz*. Certificate based and Reputation based.

In *certificate based approach* all the entities are provided with a certificate of authentication however in *reputation based model* each entity is provided with a numerical value showing the quantity of reputation various trust management approaches are considering different kinds of trust like global trust and federated Trust. In global trust each entity in the system has a specified global trust value that other entities can access. However, in federated trust the management of trust activity has been carried out across multiple and heterogeneous security domains along with the autonomous systems. Several trust based system for distributed system security have been developed like policy maker and key note.

Trust management can be defined as a unified approach to specify security policies allowing direct authorization of security critical actions. It can be defined as an activity which capture evaluate and enforce trust intentions.

### 1.2.3 Cryptography in Distributed System Security

A cryptosystem is defined as the system which transforms plaintext into cipher text using a key (*public* or *private*) as discussed by Chadwick *et al.* (2003). The process of designing a cryptosystem is called cryptology whereas the study of cryptosystem is called cryptanalysis. The fusion of cryptology and cryptanalysis is called cryptography. The cryptography is composed of two theories; and they are Symmetric key cryptography and Public key cryptography.

This provides security by massage integrity and digital signatures as discussed by Shehab *et al.* (2010). Several features of the cryptography are as follows.

1. *Confidentiality.* In this phase it is ensured that only authenticated sender and intended receiver should understand the original message. In this procedure sender encrypts message and receiver decrypt the message.

2. *End Point Authentication.* In this phase the sender and receiver are confirming the identities of each other.

3. *Message Integrity.* In this phase the sender and receiver would be ensuring that the original message was not altered during in framesets and after worlds.

Further, to ensure message integrity secured types of approach are used in cryptography such as message digest, internet checksum, Hash Function Algorithm (MDS, SHA-1), Message Authenticate Code (MAC), Digital signature and Public key certificate.

In DNA based cryptography the plain text messages are encoded in the form of DNA which is based are alphabet of short oligonucleotide sequences. The use of DNA improves the compact storage media and an externally small amount of DNA suffices even for a big one time pads.

In the present thesis we propose the development of two new security frameworks for distributed system. In the first proposed model for ensuring the trust among entities using trust values to secure the communication of trust value by using the cryptography approach. The fuzzy logic has been utilized for trust value computation. In the second model we have developed a trust based security systems for ensuring distributed system security integrated with DNA based cryptography approach.

## 1.3 PROBLEM STATEMENT

Preserving security and privacy is a challenging issue in distributed systems. This proposal makes a step forward in solving this issue by proposing a generic framework for multi-factor authentication to protect services and resources from unauthorized use. The authentication would be based on password, smart card, and biometrics. Proposed framework would not only demonstrate how to obtain secure multi-factor authentication, but also would be addressing several prominent issues of biometric authentication in distributed systems (*e.g.* client privacy and error tolerance). The model would also be satisfying the Trust Based Model properties. The major features include, identification of basic security parameters, definition and implementation of security parameters and their interrelationships, identification of biometric security issues and integration of trust based model.

## 1.4 RESEARCH OBJECTIVES AND MAJOR CONTRIBUTIONS

The objectives of the proposed research work is to enhance the security of the distributed system by proposing two new models of security framework based on the *Computational Intelligence* and *DNA Cryptography* approach. The major contributions of the thesis are as follows.

1. Computational intelligence approach is used to design a trust based system security framework. Fuzzy logic is used to approximate the trust values of entities in the proposed system.

2. DNA cryptography is used to enhance the robustness of designed system. DNA concept is used for the decryption and encryption.

## 1.5 THESIS ORGANIZATION

The thesis has been organized into seven chapters. The Chapter 1 is the introduction of the proposed work that includes the basics of distributed systems, security issues and their biometric extensions. This includes the research objectives and major contribution and thesis organization.

Chapter 2 gives the critical review of the proposed work including authentication based security approaches, trust based security approaches, access control and cryptography based approaches, policy, pattern and quorum based security approaches.

Chapter 3 introduces the basic concepts of distributed system security. Various security and legislation standards are explained in this chapter. This also includes the security services. Different cryptography issues are well discussed.

Chapter 4 introduces the application of cryptography in distributed systems. This chapter includes public key cryptosystem, RSA algorithm, Diffie-Hellman Exchange Algorithm are well discussed.

Chapter 5 includes the proposed model of trust based distributed system using soft computing approaches. Fuzzy logic theory has been used to generate the trust values. Space and time line diagram for mutual authentication has been carried out.

Chapter 6 introduces the DNA cryptography application in distributed system security.

Chapter 7 is the conclusion and future scope of the research work carried out in this thesis.

# CHAPTER 2

# LITERATURE REVIEW ON SECURITY IN DISTRIBUTED SYSTEM

In this chapter different approaches and theories related to distributed systems security are reviewed. The chapter has been divided into three sections. Section 2.1 introduces the security approaches of the distributed systems. Section 2.2 introduces the security issues and challenges and the security issues are summarized in Section 2.3.

As discussed by Pallickara *et al.* (2007) and Shehab *et al.* (2010), the important research issues in distributed systems are security approaches. The different components of distributed system security are like authentication, encryption, authorization and system protection as commented by Anderson (2010). On the other hand the security management environment (Bai (2008)) which used to be based on single authority systems is now used with groups shared responsibilities. The Distributed System access control mechanisms have been studied by Koshutanskai (2009) and the role based access mechanisms have been discussed in detail by Chadwick *et al.* (2003).

For expressive economy the term security is used to represent both its traditional meaning as well as those notions carried by the term privacy as commented by Seamons and Winsbotough (2002). An overview of distributed system architecture is usually given and employed as a framework for subsequent analysis. The major security attacks issues such as eavesdropping, masquerading and message tempering (changing the content of the message), replaying the message and denial of services have been discussed by Oppliger *et al.* (1999).

## 2.1 DISTRIBUTED SYSTEMS SECURITY APPROACHES

There are various approaches which address the security of a distributed system. These approaches are based on authentication, trust, access control, cryptography techniques etc. In this section we briefly discuss some of the approaches.

### 2.1.1 Security Based on Authentication

Shehab *et al.* (2010) have proposed a path authentication technique. On demand path discovery algorithm has been investigated to make capable domains to securely produce paths in the collaborative environment. Pallickara *et al.* (2007) produced a transport approach for tracking the availability of entities in distributed systems.

As discussed by Xiaoyong *et al.* (2011), heterogeneous distributed systems are extremely useful in different applications, like stock quote update systems, electronic transaction processing systems, which need a highly efficient amalgam of authentication, integrity and confidentiality. Xiaoyong *et al.* (2011) have also developed a meaningful security driven scheduling architecture. This approach has been developed for Direct Acyclic Graph (DAG). The developed concept dynamically evaluates the trust of each node.

The authentication of remote client is a critical research area in the distributed systems. A three factor based authentication technique has been proposed by Huang *et al.* (2011) which extends the earlier used two-factor authentication technique for ensuring the client privacy effectively in distributed systems. Three important factors used to develop this approach are password, smart card and biometrics.

Different concepts of the security in distributed systems have been explained by Vieira *et al.* (2010). This includes user authentication using passwords, digital certificates and confidentiality in data transmission. Authentication servers are playing a big role in distributed computing systems that have been discussed by Gollmann *et al.* (1993). Major design issues are the cryptographic algorithms, synchronization and amount of trust. A secured password based authentication with a trusted third party is proposed by Seung and Souhan (2006). A well-known authentication protocol, called Kerber OS is the base of this approach.

### 2.1.2 Security Approaches Based on Trust

Li and Singhal (2007) have developed a trust based model for applications like P2P systems. Trust models plays crucial role in the production of security systems in distributed applications. An extended D-S theory based trust model (ExDSTM) has been developed and implemented by Jiang *et al.* (2012). Other D-S theory models are implemented by Huang *et al.* (2010), Wang and Sun (2009) and Yu and Singh (2002).

A dynamic and context sensitive trust based security mechanism has been proposed and implemented by Ding *et al.* (2012). A risk management has been integrated into security by using a trust model as proposed by Lin and Varadharaan

(2006). This model shows that the risk management can be applied to maximize the utilization of the distributed systems. The model has the utility to evaluate the trust, also. A trust based security mechanism has been developed by Feigonbaum *et al.* (1999).

### 2.1.3 Security Approaches Based on Security

A device level system has been developed in the framework of security control and proposed by Xu *et al.* (2003). Public key cryptography, software agents and XML binding technologies are used to model this method.

Secure distributed systems use various techniques, like Public Key Infrastructure (PKI) and Role Based Access Control (RBAC). RBAC approach has been used to develop authentication based on Public Key Certificates (PKC) as proposed and implemented by Chang-Ji *et al.* (2003).

### 2.1.4 Security Approaches Based on Policy

Hamdi and Mosbah (2009) have developed a policy based distributed system security mechanism. This method gives modular security policies and independent of underlying system. This approach is based on domain-specific language for verification, specification along with the implementation of distributed system security policies and approaches. Hamdi *et al.* (2007) discussed the real integration of security policies with distributed systems. The security policies are configured manually and enforced to the distributed system automatically. Yao and Fidelis (2003) have developed a policy based mechanism.

### 2.1.5 Security Approaches Based on Patterns

Different kinds of security patterns for distributed system security are reported by Uzunov *et al.* (2012). Different pattern based security methodologies are well discussed and their maturity and appropriateness are evaluated.

### 2.1.6 Security Approaches Based on Quorum

As discussed by Zhou *et al.* (2010), quorum based security systems are extremely used in solving the problem of data consistency in distributed fault-tolerant systems. The Intrusion – Tolerance Quorum System [ITOS] of hybrid time model based on Trust Timely Computing Base (TTCB) has been proposed and implemented by Zhou *et al.* (2010).

A role based access control model has been developed by Tomoya and Makoto (2000). The Role Ordering (RO) schedulers are introduced along with concurrency control based on significance of roles assigned to the transactions.

### 2.1.7 Other Approaches for Distributed System Security

A mobile agent based security model has been proposed by Qi and Yu (2001). Explanation and analysis of the strength of security and various threats are explained. The ability of the system to detect the illegal behaviours and fight back in intrusion with counter measures is called self protection. A methodology for assessing, implementing and evaluating the self-protected system has been investigated by Palma *et al.* (2012).

The integration between security and privacy for distributed system security has been explained by Hau *et al.* (2005). The implementation of distributed security systems is optimized. Genetic algorithms have been used for the purpose as reported by Bykoyy *et al.* (2008).

A security heterogeneity method for scheduling model in the distributed system has been proposed by Xie and Qin (2007). An excellent heuristics scheduling algorithm has been implemented which strives to maximize probability that all tasks are carried out without any risk associated with attack.

Cappello *et al.* (2005) have implemented Extreme Web architecture with computing ability in a large scale distributed systems. The architecture of the system and parallel programming paradigms are explained here. A proposal for secure transaction in mobile system based on delegate object model has been proposed by Shenbagavadivu and Savithri (2012). The challenging issue of distributed nature are focused in modern computer systems.

The RAIN technology is explained by Bohossian *et al.* (2001), which is research collaboration between Caltech and NASA-JPL on distributed computing and data storage systems for future borne missions. Several solutions of concept applications are proposed, like extremely available web server, video server, distributed check pointing system.

Legal Information Flow (LIF) scheduler is implemented by Enokido and Takizawa (2007) for the synchronization of transactions to prohibit illegal information flows. An incremental progressive exposure approach and secure service discovery have been developed by Vhoi *et al.* (2011).

One another important research area is building secure P2P file sharing system. A powerful adversary model has been implemented by Di Piero *et al.* (2007) for designing a threat adaptive secure file sharing system. A CORBA based open authentication model security service specification has been developed by Chang *et al.* (2002). Bovoselov *et al.* (2007) has been proposed the security of information transmission over networks in distributed system. Zhao and Thomas (2009) discussed the secure functions in considering two models of non-repudiation protocols which are specified using the Markovian Process Algebra PEPA.

Wietrzyk *et al.* (2001) designed a model which provides support for distributed advanced workflow transactions. For the purpose of modelling security protocols in distributed systems, UML2 have been used by Zhou (2012).

## 2.2 ISSUES AND CHALLENGES IN DISTRIBUTED SYSTEM SECURITY

We mention below some issues and challenges in the development of secure distributed systems.

1. Development of approach which approximate the security level in a system.
2. Inspecting the system security.
3. Proposing the security metrics.
4. Combining the approaches like Cryptography etc. for secure distributed data communication.
5. Use of middle ware in distributed system security.
6. Use of web services in security purposes.

## 2.3 COMPARISON BETWEEN THE OLD MODELS AND THE PROPOSED MODEL

The comparison between the old models and proposed model is given below in Table 2.1:

| S. No. | Old Models | Proposed Model |
|---|---|---|
| 1. | Security policies are manually configured in model proposed by Hamdi and Mosbah (2009). | The Security policies are automatically configured in the proposed model by FKBS implementation. |
| 2. | The authentic based security model proposed by Shehab *et al.* (2010) uses manual configuration of entities. | The Proposed model automatically frames the status of entities using different trust values. |
| 3. | The Client privacy is not well implemented in the model proposed by the Theang *et al.* (2011). | The client privacy is maintained by Reputation factor in the proposed model. |
| 4. | Seung and Souhan (2006) has proposed model that uses password based authentication which is poor in the point of robust security. | In the proposed model a secure system has been developed using Reputation Factor and Trust Values. |

**TABLE 2.1 COMPARISION OF MODELS**

## 2.4 MERITS AND DEMERITS OF THE PROPOSED MODEL

The following are the merits and demerits of the proposed model:-

### 2.4.1 Merits of the Proposed Model

1. The proposed model uses modern artificial intelligence approaches to develop a robust security mechanism.
2. The proposed model uses the DNA cryptography approach that is provide multiuser architecture.
3. The uncertainty of different parameters is handled using the fuzzy logic.

### 2.4.2 Demerits of the Proposed Model

The computational time complexity of the proposed system is high compared to the other models but security degree is higher than the other models which is the major objective of the proposed system.

## 2.5 BRIEF SUMMARIZATION OF THE DEVELOPMENT OF DISTRIBUTED SYSTEMS WITH VARIOUS SECURITY TECHNIQUES

Many developments like authentication, access control, cryptographic techniques, trust based models, quorum based system etc. leads to have secure and trusted distributed systems. TABLE 2.2 briefs such type of developments.

| S. No. | Category | Focus | Reference |
|---|---|---|---|
| 1. | Authentication Based Approaches | Path authentication technique | Shehab *et al* (2010) |
| | | Security driven scheduling architecture | Xiaoyong *et al* (2011) |
| | | Remote Client authentication | Huang *et al* (2011) |
| | | Passwords, digital certificates and confidentiality | Vieira *et al* (2010), Seung and Souhan (2006) |
| | | Cryptography in authentication servers | Gollmann *et al* (1993), Xu *et al* (2003), Chang-Ji *et al* (2003) |
| 2. | Trust based security | Risk management | Lin and Vardharajan (2006) |
| | | P2P System | Li and Singhal (2007) |
| | | Extended D-S theory based model | Jiang *et al* (2012), Huang *et al* (2010), Wang and Sun (2009), Yu and Singh (2002) |
| | | Context sensitive trust model | Ding *et al* (2012) |

Table Contd. ……….

| | | | |
|---|---|---|---|
| 3. | Policy based security | Modular security policies | Hamdi and Mosbah (2009), Hamdi *et al* (2007) |
| 4. | Pattern based security | Security pattern for distributed systems | Uzunov *et al* (2012) |
| 5. | Quorum based security | Distributed fault tolerance system | Zhou *et al* (2010) |
| 6. | Other techniques | Mobile agent based system | Qi and Yu (2001) |
| | | Genetic Algorithm based | Bykoyy *et al* (2008) |
| | | X-Tron Web Architecture | Cappello *et al* (2005) |
| | | RAIN Technology | Bohossian *et al* (2001) |
| | | LIF Scheduler | Enokido and Takizawa (2007) |

**TABLE 2.2 SUMMARIES OF MODELS PROPOSED**

# CHAPTER 3

# BASIC CONCEPTS OF DISTRIBUTED SYSTEMS SECURITY AND FUZZY RULE BASED SYSTEMS

## 3.1 INTRODUCTION TO SECURITY

The security in communication networks as discussed by Pallickara *et al.* (2007) is an important research issue. In the recent past the business practices have been very much affected by data communication and e-commerce security (Vieira *et al.* (2010)) issues with the industries facing new security threats. Therefore, it is important to further strengthen the security of data transport and distribution.

The reasons behind to develop a robust security for network are as follow:

1. Security may results into the financial losses.
2. The sensitive information transferred across the Internet or Intranet must be secured.
3. The effective Information Security System must be developed by the business organizations.

## 3.2 SECURITY STANDARDS

It is important to carry out a detailed risk assessment to determine the nature and extent of important threats. This always leads to the achievement of security level that is appropriate and acceptable. The International Organization for Standardization (ISO)/ International Electro-Technical Commission (IEC) 17799 (2000) information technology as discussed by Li and Singhal (2007) is the set of information Security Management that is responsible for providing and appropriate security policies and regular auditing of systems. This standard provides specification for building and operating information security management system. It provides certificate of organizations that confirm different dimensions of security. Organizations in UK must conform to the data protection Act of 1998.

## 3.3 OPEN SYSTEM INTERCONNECT (OSI) MODEL

The Customer's Enterprise Architecture (Koshutanskai (2009)) for security is developed using Open System Interconnect (OSI) networking model. This framework has solution for security for all kind of enterprise infrastructure i.e. basic building block of any organization. In OSI model the security implementation has been carried out at different levels. Further, the security is extended into procedures and policies that support business driven goals. The security of information system plays vital role in the financial and business objectives of an organization.

### 3.3.1 Various Layers of OSI Model

1. **Physical Layer.** In this layer the cable plant wing and telecommunication infrastructure are protected using security framework. Redundant power and Wide Area Network (WAN) Connections are used to protect the physical layers. The

physical hardware in network closets, server farms are also included in the physical hardware security. Locks, alarms on entrances, climate controls and access to data centre are major building block of protection.

2. **Data link and network layers.** This layer performs numbers of technologies are used to protect the communication infrastructure. The Virtual Private Network is used to secure information by encryption method and encrypted tunnels through networks are used. The data traffic flowing over the wire is inspected by intrusion detection system on bit stream pattern. This indicates attacks or malicious intent. The suspicious pattern is identified at the hot machine using Network Interface Card (NIC) using host inclusion detection system. Malicious codes resulting into viruses are identified through virus scanning.
3. **Network and Transport layers.** It closely inspects the traffic to ensure the use of firewalls for packet entry and leaving network. The Routers used for filtering access control lists, Internet Protocol Packets traffic from going to systems are not required.
4. **Session layer.** Number of tools and technique are used to protect the system. Some of these are policies for system Management for example; handling the operating system keeping patch levels revision of operating system upto date *etc*.
5. **Presentation and Application layers.** On this layer the user accounts are managed by accessing the control networks, system and applications. Virus scanning applications are used to scan different kind of memories for malicious code.

## 3.4 SECURITY ATTACKS

An attack is an identified as an action that comprises the security of information system owned by an organization (Seung and Souhan (2006)); this is an intelligent act that is a deliberate attempt to evade security services and leads to violation of security policies in a system. There are two kinds of security attacks identified as *Passive attack* and *Active attack.*

### 3.4.1 Passive attacks

In *passive attack* the attacker only eavesdrops the communication system he may read messages and suppose to see and monitor network traffic but he does not alter messages. The meaning of passive is that the attacker does not tried to carryout modification to the data the information from the system can be used in passive attack and it does not affect the system resources. The information is collected by the attacker aiming to get the information that is in transit. Following are the passive attacks discussed below.

1. **Release of message contents.** In this attack the attacker acquires the knowledge of the confidential message that is being transmitted from sender to receiver. The email message and telephone conversation can be trapped by the attacker and this leads to the release of message content in between sender and receiver.
2. **Traffic analysis.** Traffic analysis is the act of intercepting and examining messages for acquiring information from patterns in communications. This analysis can be carried out in encrypted and decrypted messages also this can preferred in the context of military intelligence and counter intelligence and also concerned in security. An attacker acquires relevant information by monitoring the frequency

and timing of network packets. The Hidden Markov Model can be used to study the time between the keystrokes for analyzing.

3. **Brute force attack.** In this attack all the possible keys are tried decision exhaustive search attack that attempt to break a cipher by trying all possible keys.
4. **Algebraic attack.** In this attack the cipher is written as a system of equations and is solved for the keys which involves the following:

   i) Showing operations as a system equation.
   ii) Substituting known data from some of the variable and solving fro keys. This makes the attack impractical as a combination of sheer size of the systems equations.

Solving nonlinear system of equation is much harder. So cipher designing ensure to make their cipher highly nonlinear.

5. **Countermeasure for traffic analysis**. The channel can be masked by dummy traffic where no actual messages are being sent i.e. similar to encrypted traffic that maintains bandwidth usages constraints.

### 3.4.2 Active attacks

The unauthorized changes in message content have been made during active attack. There are several active attacks which are discussed below:

1. **Masquerade attacks.** In this attack a computer takes a false identity leading to acquire or modifying information. This takes place when one entity pretends to be another entity in the network. In this case the authentication sequence can be captured or replayed often a valid sequence authentication.

2. **Message  replay attacks.** The message replay is the reuse of transmitted data at a later time than actually intended frequency. For instance, in transferring funds from a bank account to another the data is captured and replayed by the attacker. It is basically the retransmission of message to execute different kind of things.
3. **Message modification attacks.** This includes the change in the packet header address for the purpose of directly and unintended destination. This attack is related to the modification of content to produce an unauthorized effect.
4. **Denial-of-service attacks.** A denial-of-service attack is related to the activity in which a particular resource denied to carry out its specified task for example a flood of message is caused in the traffic network. Another example is the execution of the rapid and reputed request to the web server. This type of attack is reported in internet connected services. This attack is wastage of network resources in real terms.
5. **Distributed Denial-of-Service (DDoS) attacks.** In this attack the attacker uses any of the computer for the purpose of attack in this case the attacker may take the full control of your computer to carryout unauthorized activity as discussed by Gollmann *et al.* (1993). There are number of effective ways to prevents DDoS but few steps can be taken to overcome with this attack.

   (i) Installing and maintaining antivirus software.

   (ii) Installing a firewall and configuring it to restrict traffic transactions.

   (iii) Apply good security practices for distributing email address and for managing unwanted traffic.

Moreover, the following symptoms could indicate the execution of DDoS:

1. Slow network performance.
2. Unavailability of a particular website.

3. Inability to access any website.
4. Dramatic enhancement in the amount of spam received in the account.

## 3.5 SECURITY SERVICES

Security services (Chang-Ji *et al.* (2003)) are implemented to increase the security of network. Security policies and mechanisms are developed using security services. There are five types of security services:

### 3.5.1 Authentication

Authentication is the process in which the entity that sends or receives the message is the one it is actually pretending to be there are two types of authentication that are used in network security:

1. **Peer entity authentication:** This is used in combination with logical connection providing confidence in the identification of entities that is connected.

2. **Data origin authentication:** This maintain authentication in connectionless transfer of data in which source of received data is as claimed.

### 3.5.2 Access Control

This prevents the unauthorized access of resources. This can also be defined as the ability to permit ordinary the use of something by someone.

### 3.5.3 Data Confidentiality

This ensures the safe delivery of data from origin to destination. There are four types of data confidentiality as expressed below:

1. **Connection Confidentiality:** This is related to the protection of all user data on a connection.
2. **Connectionless Confidentiality:** This is related to the protection of user data in a single data block.
3. **Selective-Field Confidentiality:** This is related to the protection of selected fields in the user data on a connection.
4. **Traffic Flow Confidentiality:** This is related to the protection of data that is derived from observation of traffic movement.

### 3.5.4 Data Integrity

The data integrity assures that the received data is same as delivered by authorized entity that means it contain no modification or alternation in the data. The various data integrity methods are as under.

1. **Connection integrity with recovery:** This provides integrity of all user data in a connection and identifies any type of alternation in data with recovery attempted.
2. **Connection integrity without recovery:** This is same as above discussed integrity but having without recovery.
3. **Selective-fields connection less integrity:** This is related to the integrity of selected fields within the data block of user data transferred over a connection.

4. **Connectionless integrity:** This is related to the integrity of single connectionless data block and can take the identification of data alternation. This also has a replay detection mechanism.

## 3.6 SECURITY MECHANISMS

The security mechanisms (Hamdi and Mosbah (2009)) are responsible for identifying any break in security and after identifying this helps to remove the security breakdown. Security mechanisms are responsible for identification of various security attacks. Different Security mechanisms are discussed in the section.

1. **Encipherment.** This is also termed as encryption. This develops the mathematical formulas, algorithms and key to convert a simple message into a complex message that is not easily understandable by the each and everyone.
2. **Digital signatures**. These are used to ensure the source and integrity of computer document. The cryptographic formats of digital signature are appended with the electronic document.
3. **Access control**. This is used to prevent unauthorized access of data and network resources.
4. **Data integrity.** This identifies that the data sent to the receiver is same as the data sent by the sender.
5. **Routing control.** Routing is the important activity to tackle with the problem of traffic congestion thus leads to removal of denial of service attack.

## 3.7 CRYPTOGRAPHY

Cryptography is based on the functioning of encryption and decryption of data which enables us to store sensitive information or transmit it across in secure network. This is a practice of hiding the information being transmitted. It is basically the combination of Mathematics and Computer Science.

### 3.7.1 Conventional Encryption Model

In this particular security model the message is set over the network with the following steps:

1. The sender takes decision to send a message over the network this is represented in any language of users' interest and known as plain text.
2. After the encryption is performed using cryptographic keys and plain text is converting into cipher text.
3. Finally the cipher text message is  send over the network on the communicated channel.
4. After receiving the message on the receiver the cipher text is converted into plain text using some set of keys.

This security framework is of two types, namely symmetric cryptographic model and Asymmetric cryptographic model. In symmetric cryptographic model the encryption and decryption are carried out using same key that is called shared key. The shared key is the unique key for a specific session and is only known to sender and receiver. In the asymmetric model public key of sender or private key of receiver is used for encryption and similarly they receive decrypts the information all users of the networks know the public key but a private key is only known to a particular user.

### 3.7.2 Kinds of Ciphers

There are many types of ciphers are discussed as follows:

1. **Substitution cipher.** In substitution cipher the changes are made some letters in the plain text to convert into cipher text.

2. **Caesar cipher.** This is also known as the shift cipher and this is most widely used encryption technique. This method is named after Julius Caesar who used this method to communicate with their general. The following example can be quoted.

The plaintext is given as:

Plaintext: L U C K N O W
And the key is 3. So the ciphertext comes out as:
Ciphertext: O X F N Q R Z

The process of deciphering is just the reverse of this process in which the letters are identified by decreasing degree the modular arithmetic method can also be used for encryption and decryption. This is mathematically expressed as

$$E(y) = (y + m) \bmod 26.$$
$$D(y) = (y - m) \bmod 26.$$

In this scheme a = 0, b = 1, ……., z = 25 are the modulo number identification.

The Caesar cipher can be easily broken under two types, namely, monoalphabetic and playfair cipher. In monoalphabetic cipher one letter of the alphabet is substituted with another letter of alphabet Caesar ciphers are best examples of Monoalphabetic cipher. The playfair is also named as playfair square which is manual symmetric encryption technique invented in 1854.

## 3.8 DATA SECURITY PROTOCOLS

The data security protocols are developed to maintain security against various kinds of attacks. Eavesdropping, man-in-the-middle attacks, repudiation, message alternation *etc*. are some of the examples of the data security protocols. The authentication, authorization and encryption are the mechanism of data security protocols, less computing power is required by private key encryption as compared to public key encryption. Some time public key approaches are used to distribute the private keys. One of the private approaches is Kerberos developed at Massachusetts Institute of Technology (MIT), USA. This contain central security server which is trusted purely by everybody. The public key approaches are currently being used in security models that include Rivest-Shamir-Adleman (RSA) algorithms, elliptic curve cryptography.

## 3.9 CORBA ARCHITECTURE

This is developed by the Object Management Group (OMG). This architecture is based on the statement that the developers are not agreed upon common languages like Java and any common operating system. So it is required to have a security layer in between layers and operating system.

### 3.9.1 General Architecture

CORBA application communicates with objects. An Interface Definition Language (IDL) is developed for each kind of objects. The information is passed to the IDL that makes the broker to have correct objects. The scripting language has been developed using C, C++, Java, COBOL, Small Talk, ADA, LISP, Python etc.

There are many security aspects that are mainly implemented by CORBA architecture. Following are the different steps that show the functioning of CORBA.

1. **Identification and Authentication.** This means the identification of real user that who is sending or receiving the information. In this system the user is authorized by a password system for the purpose of accountability for access to objects with different permissions for controlling access to different objects message.
2. **Authorization and Access control.** When the users are authenticated the applications should have credential to access other objects through ORB. This is implemented on the behalf of security policy for granting the access.
3. **Security auditing.** The Auditing is an important component of security which enables the administrator to identify intrusions and other security anomaly. There are two kind of auditing in CORBA, System and Application. The System policies include the event logging like authentication of principles changes in privileges etc. All the applications are automatically enforced by all the audits.
4. **Non-Repudiation.** This sets the accountability of all actions carried out. CORBA provides non-repudiation of creation receipt creation of messages etc.
5. **Administration:** The Domains are used to administer the security in CORBA. There are three kinds of security domains - Security policy domain, security

environment domain and security technology domain. The management interfaces are not provided by the administration. This also maintains and establishes security services.

### 3.9.2 COM+

This is the architecture used for distributed system security in Microsoft COM+ Architecture. It is regarded as the next generation of distributed architecture which provides automatic security. The developer can leave security out of the components they create the writing and maintain the code is easier and designing of security at higher level throughout the entire application has been maintained.

## 3.10 FUZZY LOGIC

The Fuzzy logic (Zadeh (1965)) deals with the uncertainty in human decision making. The human mind has a remarkable capability to reason and make decisions in an environment of uncertainty, imprecision, partial truth. The principal objective of fuzzy logic is formalization/mechanization of this capability of human mind. In other words, the fuzzy quantifies the linguistic value of the phrases like 'he is a *young* person' and 'this building is *very high*'. Some of the important features of the fuzzy logic are deals with approximate reasoning and multi-valued logic, represent and to process the linguistic information and subjective attributes, extension of Boolean Crisp Logic to deal with the concept of partial truth.

Normally crisp sets are represented as expressed in Figure 3.1 given below.

Tall={165, 170,175, 180, 185}

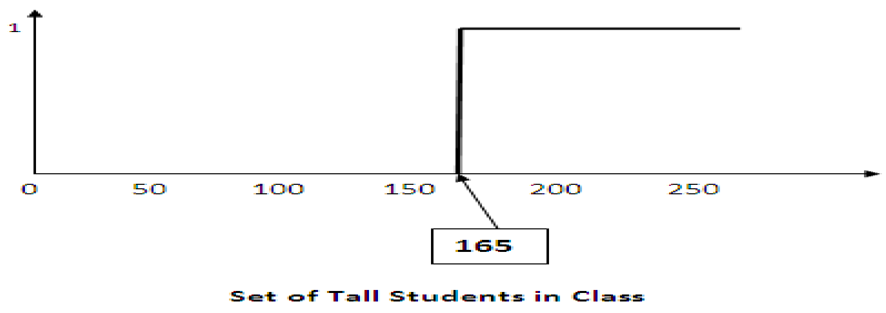


Figure 3.1 Crisp set representation

Now similarly the same example has been represented by the fuzzy set. The examples are as follows:

Tall= {(152, 0.2), (160, 0.7), (164, 0.9), (165, 1), (170, 1), (175, 1)}

Tall=0.2/152 + 0.7/160 + 0.9/164 + 1/165 + 1/170 + 1/175

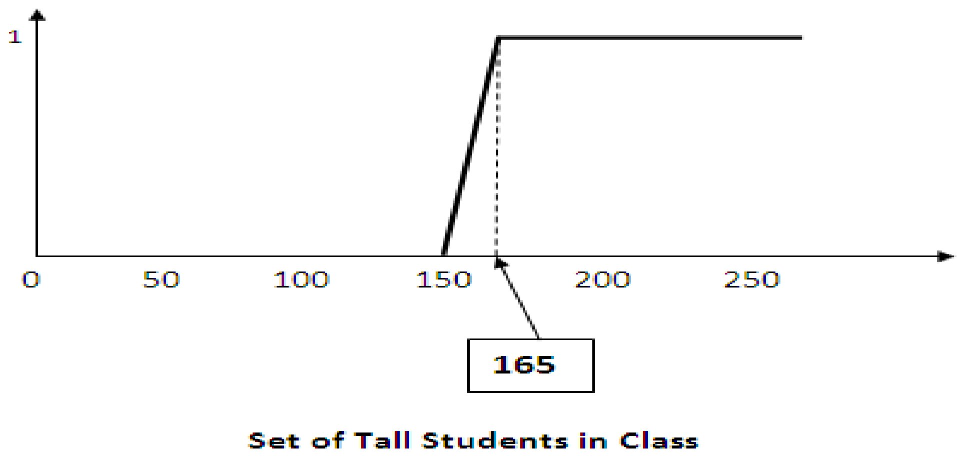


Figure 3.2 Fuzzy set representation

### 3.10.1 Fuzzy Set Representation Method

Fuzzy sets are expressed by two methods - discrete and continuous which are defined below.

**Discrete method** If *A* is a fuzzy set then following is the discrete method representation.

$$A = \mu_A(x_1)/(x_1) + \mu_A(x_2)/(x_2) + \cdots\cdots + \mu_A(x_1)/(x_1) + \cdots\cdots \qquad (3.1)$$

The examples of this representation are as follows;

Driving Speed on Highway

Fast speed

F= 0.6/80 + 0.8/90 + 1.0/100 + 1.0/110 + 1.0/120

Medium speed

M=0.6/50 + 0.8/60 + 1.0/70 + 1.0/80 + 0.8/90 + 0.4/100

**Continuous method** The membership function in the continuous method is expressed as follows:

$$A = \int_{x\epsilon X} \frac{\mu_A(x)}{x} \qquad (3.2)$$

Here are few examples of the membership function representation. The Figure 3.3 is the membership function of linguistic variable temperature.

Temperature={Cold, Cool, Warm, Hot}

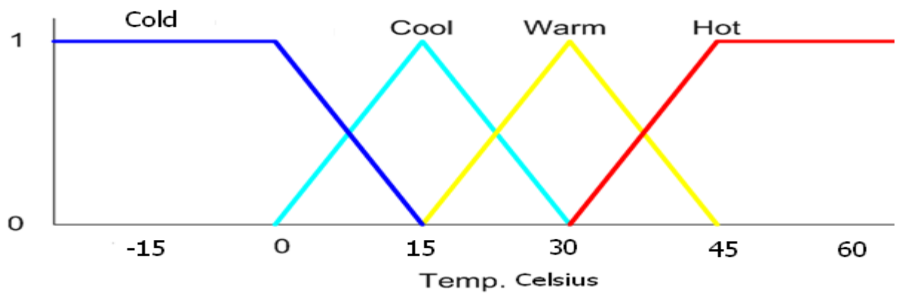


Figure 3.3 Membership Function of Temperature

**Fuzzy set operations**

The union operation is mathematically represented as follow

$$\forall_x \in U, \mu_{AUB}(x) = \max(\mu_A(x), \mu_B(x))$$

(3.3)

Example:

Fast and Medium Speeds at Highway

F=0/50 + 0.2/60 + 0.4/70 + 0.6/80 + 0.8/90 + 1.0/100 + 1.0/110 + 1.0/120

M=0.6/50 + 0.8/60 + 1.0/70 + 1.0/80 + 0.8/90 + 0.4/100 + 0.1/110 + 0/120

FUM=0.6/50 + 0.8/60 + 1.0/70 + 1.0/80 + 0.8/90 + 1.0/100 + 1.0/110 + 1.0/120

The mathematical definition of intersection operation is given below.

$$\forall_x \in U, \mu_{A \cap B}(x) = \min(\mu_A(x), \mu_B(x)) \qquad (3.4)$$

Example: Fast and medium speeds at highway

F= 0/50 + 0.2/60 + 0.4/70 + 0.6/80 + 0.8/90 + 1.0/100+1.0/110+1.0/120

M= 0.6/50 + 0.8/60 + 1.0/70 + 1.0/80 + 0.8/90 + 0.4/100 + 0.1/110 + 0/120

FUM= 0/50 + 0.2/60 + 0.4/70 + 0.6/80 + 0.8/90 + 0.4/100 + 0.1/110 + 0/120

The diagrammatic representations of the operations carried out on fuzzy sets are given below.

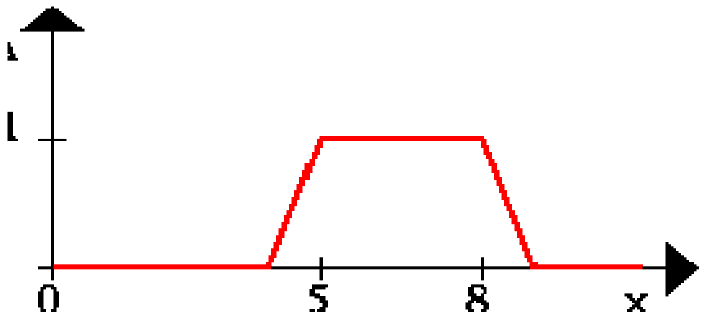


Figure 3.4 Fuzzy Membership Function A

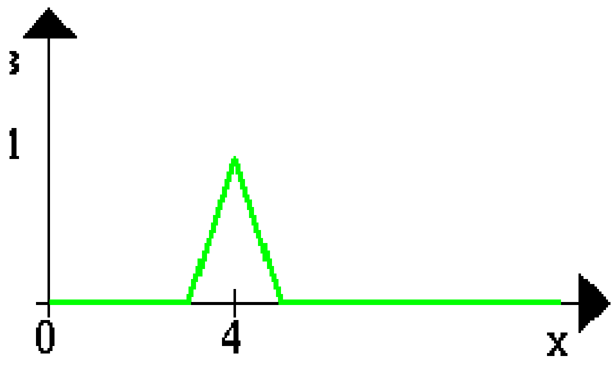


Figure 3.5 Fuzzy Membership Function B

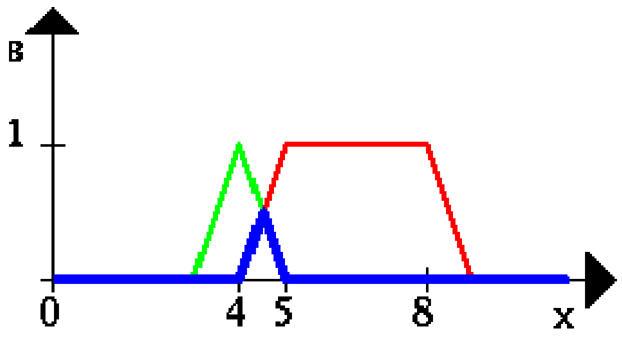


Figure 3.6 Intersection operation of A and B

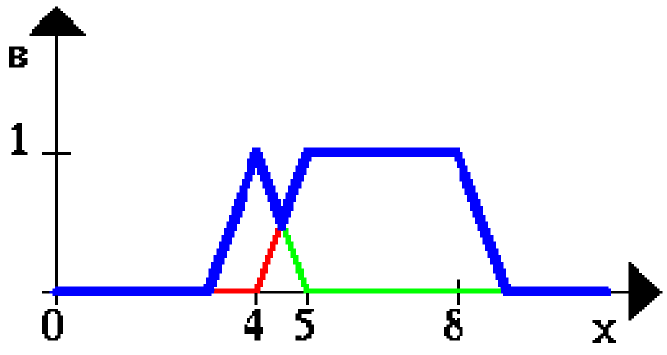


Figure 3.7 Union operation of A and B

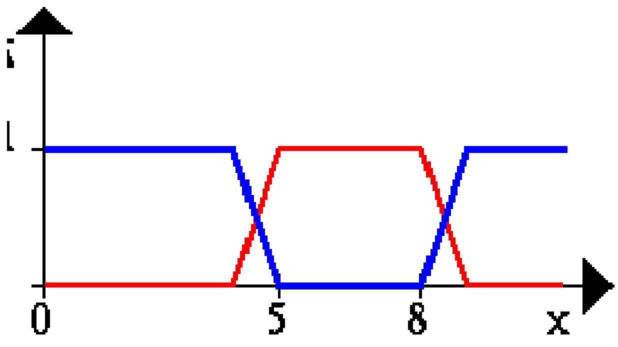


Figure 3.8 Complement operation of A and B

**Support on fuzzy sets.** The support of a fuzzy set is mathematically defined as follows:

$$Sup\ p\ (F) = \{u \mid u \in U \text{ and } \mu_F(u) > 0\}. \qquad (3.5)$$

Example:

M = 0/10 + 0/20 + 0.6/50 + 0.8/60 + 1.0/70 + 1.0/80 + 0.8/90 + 0.4/100

Support (M) = 0.6/50 + 0.8/60 + 1.0/70 + 1.0/80 + 0.8/90 + 0.4/100

**Kernel on fuzzy set** The kernel of the fuzzy set is defined given below.

$$ker\ (F) = \{u \mid u \in U \text{ and } \mu_F(u) = 1\}. \qquad (3.6)$$

Example:

M = 0/10 + 0/20 + 0.6/50 + 0.8/60 + 1.0/70 + 1.0/80 + 0.8/90 + 0.4/100

*ker* (M) = 1.0/70 + 1.0/80

**Alpha Cut** Strong and weak alpha cut are two examples of the alpha cut. These are mathematically defined as follows:

**Strong alpha cut**

$$F_{\alpha}+ = \{u \mid u \in U \text{ and } \mu_F(u) > \alpha\}. \qquad (3.7)$$

**Weak alpha cut**

$$F_{\alpha} = \{u \mid u \in U \text{ and } \mu_F(u) \geq \alpha\}. \qquad (3.8)$$

Example:

M = 0/10 + 0/20 + 0.6/50 + 0.8/60 + 1.0/70 + 1.0/80 + 0.8/90 + 0.4/100

If $\alpha = 0.6$

Then Strong α Cut = 0.8/60 + 1.0/70 + 1.0/80 + 0.8/90

Weak α Cut = 0.6/50 + 0.8/60 + 1.0/70 + 1.0/80 + 0.8/90

### 3.10.2 Fuzzy Rule Based Systems

These are the systems used in different kind of applications and also integrated in distributed system security. This system has the following components.

1. Knowledge base: This is the repository of the knowledge about related problem. This consists of Data Base and Rule Base. Data Base is the collection of membership functions and Rule base is the collection of the number of rules representing the knowledge.

2. Inference engine is the component that is responsible for making decisions.

3. Fuzzification interface is responsible converting the crisp information into fuzzy.

4. Defuzzification interface is responsible for the conversion of fuzzy output into crisp output.

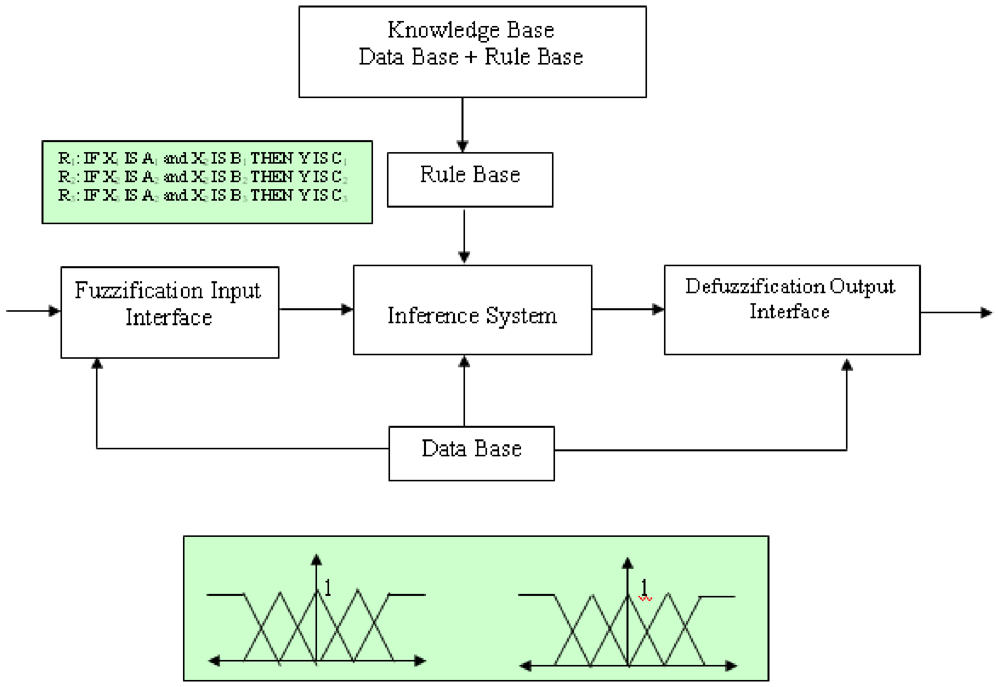

Knowledge Base
Data Base + Rule Base
$R_1$ : IF $X_1$ IS $A_1$ and $X_2$ IS $B_1$ THEN Y IS $C_1$
$R_2$ : IF $X_2$ IS $A_2$ and $X_2$ IS $B_2$ THEN Y IS $C_2$
$R_3$ : IF $X_3$ IS $A_2$ and $X_2$ IS $B_3$ THEN Y IS $C_3$
Rule Base
Fuzzification Input Interface
Inference System
Defuzzification Output Interface
Data Base
1
1

Figure 3.9 Fuzzy Rule Based Systems

## 3.11 TRUST BASED SECURITY SYSTEMS

The trust is defined (Lange and Oshima (1998)) as a set of assertions that a principal held with regard to another principal. An assertion may either be positive or negative, and in the latter case, we specifically call it distrust. It may be noted that distrust is different from the absence of trust, which merely indicates lack of knowledge. Depending on the interpretation, an assertion may be treated as a principal's belief about other principals, or a weaker interpretation may simply treat an assertion as one's statement about others. An assertion is often associated with a specific context, where a context is defined as the situational conditions under which an assertion is expected to be interpreted with its intended meaning. From the perspective of the framework, there is no specific format for assertions.

The trust (Fidelis) is embodied in trust statements. A trust statement is a signed credential with a trustier and a subject. The trustier is the issuer of the trust statement; the subject is the principal the trust statement concerns. A trust statement represents a trust relationship between the trustier and the subject, and is signed by the trustier. In trust statement the signature is the most important factor which solves two purposes. Firstly, it proves the authenticity of a trust statement and secondly (*cf.* Varadharajan (2005)) it indicates the explicit source of a trust statement. A signature can be classified as digital or non digital, it creates a strong relationship between the signer and the signed entity. The basic objective of a signature is to prove the consent of the signer with respect to the signed entity. Since a signature is assumed to be unforgettable which only its owner can produce, a signed trust statement identifies its trustier. The non-repudiation is an extra property in signatures. Moreover, the claiming responsibility and liability increases the trustworthiness of a recommendation. Likewise, the trustworthiness increases if the signature of a trust statement offers a non-repudiation guarantee.

The conveyance instance (Lin *et al.* (2005)) can be defined as if the trust to be conveyed that if one principal passes a trust statement to another. Such an instance is called a conveyance instance. A conveyance source (or source) is defined as the principal who transfers a trust statement in a conveyance instance, and a conveyance target (or target) is defined as the principal who receives the trust statement in a conveyance instance. A source may or may not be the trustier of the conveyed trust statement, although it is often the case that the trustier acts as the source for its own trust statements. Similarly, the target need not be the subject of the trust statement it is receiving. There are two basic principles used in the trust conveyance approach - subjective and Dynamic. In the subjective trust approach every principal has the discretionary power to make its own trust decisions, which may be based on the trust statements it believes. whereas in the later approach the trust statements is subjected to some validity conditions so that one's representing outdated knowledge will be invalidated. Besides these two principles, this imposes no further assumptions on the concept of trust. In particular, it does not force a single-minded view of trust. Instead, every principal has complete freedom to choose its trust model, which may have a definition of trust level and/or methods for computing trust. In this regard, trust statements serve as an interface to communicate with other principals. This departs sharply from other approaches which attempt to define domain-specific trust models.

The computing systems are evolving into distributed systems that interconnect competing organizations and individuals, and even countries, using high-speed global networks. The relationships among these entities are characterized by the need for competition and cooperation without a common trusted agent as discussed by Abdul-Rahman and Hailes (2000). To build such distributed systems that incorporate lack of global trust in them, it is necessary first to understand precisely what trust consists of and then to categorize it. In the present work a systematic methods has been developed for synthesizing protocols that implement a given trust specification.

Trust (Beth *et al.* (1994)) is primarily required to establish channels for secure communication. We present methods for reasoning about trusts required by various channel establishment mechanisms. Channel establishment mechanisms are commonly based on either public key encryption (PKE) or single key encryption (SKE). PKE-based mechanisms require ternary trust relationships known as authenticity trusts. SKE-based mechanisms have much larger trust requirements. Starting from the differences in trust requirements of PKE and SKE, we derive several advantages of the former over the latter. Our analyses provide insight into the trust structure and limitations of various mechanisms.

We have shown how a distributed system must provide a tree of channels at system configuration time, and this tree also represents the system's global name space. We develop polynomial-time algorithms for synthesizing name spaces so as to satisfy an a priori given set of trust specifications. We present some interesting duality results and NP-completeness results with regard to some variations of the synthesis problems. Sample runs of the polynomial-time algorithms show that small differences in trust relationships can cause substantial differences in the structure of the name spaces.

Trust requirements and the performance of channel establishment as discussed by Bierman and Cloete (2002) can be traded for each other. If channels are PKE-based, slightly increasing the trust requirements can greatly increase the performance of channel establishment. However, if channel composition is SKE-based, global trusts, which may not be satisfied in the system's name space, are required for significant improvements in performance.

### 3.11.1 Trust as a Security Concept

In information security the trust is the fundamental component as commented by Gasser *et al.* (1989). The security depends on the appropriate functions of the hardware and software, the validation of cryptographic algorithms, the validation of cryptographic protocols, etc. Even without being explicitly stated, trust is placed on every link in the chain of security for a system to be considered trusted. If any of the components in a link is broken, the security of the system would be defeated. In this respect, the theory of computer security is directly proportional to the dependable computing, wherein the notion of trust has an element of reliance in both areas. Therefore, the trust is equivalent to dependability. The system will be operated within a certain level of confidence, reliability and dependability. In cryptographic protocols the trust can also be seen implicitly. For example, the basic concept of authentication protocols is to derive a specific type of trust as a conclusion: the belief that the communicating entity is indeed the claimed principal. Depending on the details, the execution of a protocol often needs to make a number of trust assumptions on either end of the communication, e.g. the belief that the server will generate a session key of a sufficient strength, the belief that the server will not leak out confidential information, etc. The observation here is that trust is relative to specific tasks. Trusting a server for authentication does not imply that the server should be trusted for secure storage of confidential data. The purpose associated with trust must be explicitly stated.

A form of trust (Gollman (1999)) can be seen in the logic for distributed authentication which includes a construct for describing delegation of rights. They define and speak for relationship between the users A and B such that if A says any statement, we can believe that B says the same statement. This type of trust encompasses the notion of honesty. If A is trusted to speak for B, then it is believed that A will honestly say a statement that B also says. It is noted in that as well as honesty, the concept of responsibility should also be considered in delegation. Responsibility is a means of managing risks so that, for example, the possible damage and liability of an action by a delegated principal can be accounted for. This crucial observation suggests that trust has an intimate connection with risks. Further, when A and B is trusted, then every statement made by A is believed to be also made by B, including the concept of roles which allows a principal to limit its authority. A role may be defined as the name of a program, e.g. NFS server, or its class or un-trusted server. Principal A acting in role R is written as A as R. A weaker trust relationship of speaks for can be expressed as A to B as R as explained by Grandison and Sloman (2000).

Another significant modeling of trust can be seen in public key management, where the term “Trust Model” is used to describe the structure of certification authorities, recognizing that the monolithic, single-tree approach of the original X.500 is unlikely to be realized as given by Jansen (1999). The basic idea of trust in this field is narrow, referring specifically to the authority to certify keys. The use of the term trust model here could in fact be more precisely described as certification topologies. The trust in security assumes complete certainty. If a computer system is certified to be trusted at a certain evaluation level, it implies it should always function within the guarantees of that level provided correct operating procedures are followed. In logic, if a principal is trusted, it means it will always demonstrate certain expected properties, e.g. to have jurisdiction on asserting public keys for same principals. Trust in security research is taken as a binary concept.

### 3.11.2 Trust as a Sociological Concept

In contrast to the simplistic views of trust adopted in security research (Josang (2002)); trust has also been studied in a much wider context in other disciplines. In general, the word trust is often used by people in a very broad sense to mean a number of things. Its interpretation by the trusting party varies significantly, depending on past experiences, associated risks, recommendations from other parties, reputation of the trusted parties, or even cultural background. It is not always clear to every person how trust or distrust is derived in every case, and indeed, sometimes this process occurs subconsciously. For example, some people base their trust decisions strongly on trust instinct, or psychologically place more trust on people of their own race. However, there is a fairly uniform recognition among researchers that trust is a subjective measure. Given the same external conditions, people may often have a different degree of trust over the same matter. Also, trust (or symmetrically distrust) (Josang (2001)) is a particular level of the subjective probability with which an agent assesses that another agent or group of agents will perform a particular action, both before he can monitor such action and in a context in which it affects his own action.

A security application (Kwork *et al.* (2005)) may require strong absolute trust, while fuzzy trust may be preferred in e-commerce applications which may be backed by dispute resolution and compensation plans so that business between complete strangers can be carried out. Based on this premise, Fidelis advocates a different approach, centering on the notion of trust conveyance.

### 3.11.3 Validity

There are a number of techniques for expressing validity conditions as explained by Marsh (1994). X.509 specifies a coarsely grained validity period for its certificates, with the assumption of synchronized clocks at the global scale. It uses a revocation list mechanism to invalidate a certificate prior to the end of its validity period. Other work on applying tree structures to improve revocation includes OASIS uses efficient asynchronous messaging to maintain real time validity of its certificates. This is complemented by the infrastructure support for network failure detection.

For instance, the conveyance model described above does not prescribe a particular validity mechanism (Oppliger (1999)). The different validity mechanisms deliver different degrees of guarantee, and it is an application issue to determine the validity strength of its trust statements. The model however requires a validity method to follow a determinism principle. The principle is that the validity of a trust statement cannot be negated once it is guaranteed. A consequence is that the processing behavior will be deterministic with no sudden surprises. These semantics are desirable especially in a widely distributed system where network failure and partition are inevitable. As an example, a possible validity mechanism that exhibits deterministic behavior would be a simple validity period without revocation lists. The absence of revocation lists ensures that a trust statement only invalidates at the end of its period, thus the validity guarantee cannot be broken by any means. This is an example of an online mechanism, where the validity of a trust statement is maintained independently of the availability of the network. A family of online mechanisms is supported in the Policy Language.

There have been several attempts to model trust in the past. This model computes trust values based on the degree of trust of recommenders. Some of their models include protocols for updating experiences and recalculating trust values. Attempts to capture trust using subjective logic that computes an opinion value along three axes, belief, disbelief and uncertainty are computed. It presents a trust model for e-commerce, which computes trust values to include parameters such as transaction cost, transaction history, customer loyalty, etc. This model incorporates the concept of risk analysis and is based on a fuzzy logic for inferring trustworthiness. Similar to others, it is also described a protocol for maintaining trust values.

There exist many other similar attempts for different application areas. It is unlikely that a unified model will ever exist to satisfy individual needs. Instead of proposing yet another trust model, the trust conveyance model attempts to provide a framework in which these trust models may interoperate and cooperate. One of the primary reasons for defining trust models is to create a basis for participants to infer trust-related decisions. In large distributed systems, there are three difficulties with this approach. The notion of trust differs significantly depending on the nature of applications. Second, such models typically require some monitoring mechanism to ensure every participant's compliance. Distributed monitoring is however subject to operational availability of the infrastructure and general scalability problems. Third, autonomous participants may have different trust assessment schemes, which include subjective opinions and errors. It is unclear how a trust model can be enforced in the light of principal autonomy.

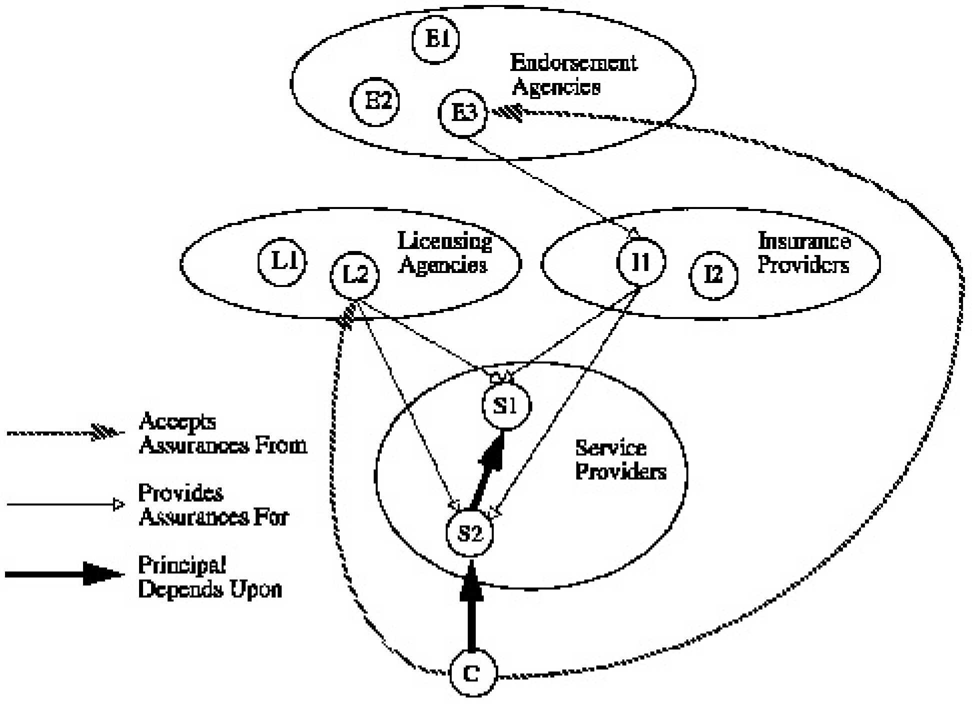


Figure 3.10 Validity Trust Based Systems (Rasmusson and Jansson (1996))

### 3.11.4 Identity

Identity adopts a key-centric approach which identifies principals by public keys. A principal may represent a person, an organization, or a computer process, etc. The key-centric treatment does not distinguish the actual entity represented by the principal, but instead insists that a principal must control (*i.e.* speak for) a public key pair. Every principal may freely generate a public key pair at anytime. The generated public key can then be used as an identifier for the principal. Global uniqueness is guaranteed by the fundamental requirement of the chosen public key cryptosystem, which ensures no collision of keys is possible, given sufficiently large entropy, *e.g.* 1024 bits. A prerequisite assumption for this key-centric approach is that every principal should exercise good safeguarding practice for its private keys, which is a typical assumption for public key cryptography. There are measures to encourage and enforce this prerequisite requirement.

A principal may control multiple keys simultaneously (Song and Hwang (2001)). It is a common practice to limit the damage of a possible compromise of a key by constraining its use. Suppose a principal has a key pair for e-mail communication and another for workstation login. If the former key pair is compromised, it would only affect its e-mail usage but cause no damage to workstation access. The same physical principal is therefore allowed to be identified by multiple public keys. Each public key is treated as a separate instance of the principal.

This key-centric approach provides a possibility for anonymity. Provided a principal generates a fresh public key on every anonymous access and, by requirement, there is no mathematical relationship between any two keys, the principal can electively hide, its identity using a new public key. It is important to note that this mechanism alone is not sufficient to prevent analysis based on linked access patterns and attacks based on collision. Public keys as principal identifiers merely provide a ready source of pseudonyms.

The trust conveyance model places two requirements on naming support. First, a conveyance target must be able to validate the authenticity of a trust statement based on the identity of the truster. Second, every principal must be uniquely identified in the system. Failure of this introduces ambiguity and prevents communication between arbitrary pairs of principals.

A possible approach to satisfy these requirements is to deploy a global naming system and couple it with a public key infrastructure. The original plan of the X.500 directory service is a prime example of this approach. The idea is to associate every principal with a hierarchical name. Association between a public key and a name is then certified by some Certification Authority (CA). There are several problems with this approach as discussed in the literature. Hierarchical namespaces are introduced to address the scalability problems associated with at namespaces. However, this in itself requires a standard hierarchy so that names can be meaningful to every principal. This is often difficult, if not infeasible, since every community will have a preferred naming structure, for intuition and convenience reasons. The partitioning of namespaces must be permanent to ensure the validity of a name. Evolution of a namespace will invalidate all of its dependent namespaces in the hierarchy. Furthermore, hierarchical namespaces require naming authorities at each level to ensure unique allocation of names. This centralized management, even scoped locally, may eventually become a problem in large-scale systems with potentially thousands of users.

A more significant problem is global key management. Because of the hierarchical nature, trusting a key binding implies trusting all the intermediary authorities along the chain to the root authority.

Breach of security at an authority will therefore have a propagating elect to all its descendants. The root of the hierarchy becomes an attractive point for attack, since breaking the root will enable an adversary to control the entire structure. This problem is largely due to the implicit assumption in X.509 where a naming hierarchy is assumed to react the trust hierarchy for key certification. This aggregates trust towards the root of the hierarchy, *i.e.* the higher up in the hierarchy, the stronger trust assumption is required. This rigid assumption precludes the dynamic nature of today's distributed applications, where trust relationships tend to be complex and constantly changing. More importantly, it forces applications to adopt a single trust structure.

The key-centric approach pioneered by modern key-oriented access control schemes presents an elegant solution. Public keys are mathematically designed to be globally unique by nature. The probability of two keys clashing is negligible. Besides that public keys do not need to be kept secret. The real value of the key centric approach is the avoidance of names. An important observation is that names are mostly for the convenience of humans. People are used to identifying others by names a practice learned from the early days of one's life. While natural for humans, names are of little value for computer systems. In an open system, the strongest guarantee is the knowledge that the remote communicating party controls a particular private key. Proving the name is a secondary action which requires a secure binding from the key to the name. The key-centric approach does not deal with names and hence eliminates the need for name management. A desirable consequence is the independence from central trusted third parties to certify the authenticity of keys. If a principal can be identified, its key will be known. This trusts naturally with the trust conveyance model, where a public key in a trust statement can both identify its truster and verify its integrity.

Although the key-centric approach solves the global naming problem, on the other hand, it introduces another problem due to its source of principal identifiers. Since by assumption, every principal may generate a fresh key pair and use the public key as its identifier, the public key is inherently anonymous. For example, if a principal is blacklisted for financial fraud, he/she may simply generate a new key pair, essentially creating a new identity, to avoid being caught.

This problem is considered a policy issue. It is up to each individual service to decide whether anonymous public keys are accepted. If a service requires persistent names, it may demand a principal to present trust instances issued by some trusted authority, e.g. the Government registrar providing a name-certification service, linking public key identifiers to names. To bind a name to a public key in an authoritative manner, the authority should typically follow rigorous procedures, identifying both the ownership of the key and the name, and possibly some additional attributes that are asserted.

### 3.11.5 Principals

There are three types of principals. A plain principal is specified as its public key. A principal group is specified as a set of public keys. A threshold principal is specified as a set of public keys, with a threshold value of the minimum number of representative  in the set.

In Group principals are conjunctions of principals. The intuition is to treat the principals in a group as a single, logical principal. This enables representation of concepts such as joint statements. A trust statement signed by a group principal is semantically identical to the same trust statement individually signed by all members of the group and aggregated together. Further, a threshold principal is a special type of group principal. While a plain group principal represents the entire set of group members, a threshold principal represents a subset of a group, with a minimum number of principals in the set. The minimum number is the threshold value, specified as an integer. The threshold construct enables the specification of threshold schemes. A common commercial threshold scheme would be that a company cheque typically requires two or more signatures for it to be valid. Moreover, the principal set for group or threshold principals may be specified literally, as shown above, or refer to a variable which will be bound during evaluation. This is useful for large groups or dynamic groups backed by databases.

The language provides the self keyword for representing the public key of the policy owner. In theory, there is no difference between a policy owner and the rest of the world a literal representation can be used to identify the policy owner. It is however sometimes useful to late-bind the policy owner at deployment rather than at specification time. This allows some degree of centralized policy management, whereby an authority may define a standard trust policy and distribute it to participating principals for enforcement. An any-key keyword is provided as a wildcard for public keys. It is intended for policies that need not consider specific trusters or subjects. For example, a policy may state any person certified by the local authentication server may log onto a workstation.

### 3.11.6 Actions

An action encapsulates computation that may be subject to policies (Varadharajan (2000)). As a motivating example, consider an access control scenario, where an access control monitor in an operating system needs to determine if a requester is allowed to read. The notion of actions is typically defined differently across applications. To satisfy these diverse needs, the language generalizes actions as parameterized predicates.

There is no built-in type system in the language. It is deemed to be an implementation and deployment issue. There are numerous choices in programming languages (*e.g.* Java, C, C++), database management systems (*e.g.* SQL, OQL/ODL), and distributed middleware (e.g. CORBA, DCOM). The type system used in a policy must be identified when it is processed. A simple type system consisting of only primitive types, including int, float, and string, will be used. Public keys will have a special primitive type public key. An action instance is an instance of an action specification. It is defined by a name and a list of parameter instances. The name refers to an action specification, and the parameter instances must match the specification. A parameter instance is given as a literal value in the value space of the parameter type.

### 3.11.7 Trust Specification

The trust specification (Sonntag and Hrmanseder (2000)) employs a similar abstraction for expressing assertions as for actions. Assertions are represented in the form of parameterized predicates. As with actions, the parameter list is optional. Some assertions are simple and narrowly scoped, and can be expressed without parameters. An example would be paid in an online purchase session. A customer who has paid for a purchase may be certified by the accounts department of the selling company, and its delivery department, based on this assertion, may then arrange for purchase dispatch. Such trusts are Boolean, i.e. only “believed" or “not believed".

It is important at this point to distinguish trust statement instances from trust specifications, which were collectively referred to as trust statements previously. A trust specification is not bound to a specific truster and subject. Only beliefs are specified. A principal instantiates a trust specification in the capacity of a truster regarding its belief concerning another principal. A concrete trust statement is referred to as a trust statement instance or simply trust instance.

### 3.11.8 Trust Relationships

The most important aspect of the language is the specification construct for trust policies as discussed by Yu and Singh (2002). Generally speaking, a trust policy defines a principal's belief about another principal, i.e. a trust relationship. The trust policy construct offers building blocks for capturing common factors of trust, including recommendation, reputation and to a certain extent, experiences.

A trust policy may serve two purposes. First, it defines conditions for trust establishment. For example, A may specify conditions that must be met before she trusts B to sell books. B may approach A to obtain the trust by presenting proofs. If A's conditions are satisfied, it establishes a trust relationship with B by creating and signing a trust instance. Second, it assists trust decision-making. Continuing the previous example, suppose C wishes to determine B's trustworthiness for selling books. It may approach A with some beliefs she holds about B. Alice may then reply to C if she thinks B is trustworthy according to her own policies. Before the syntax for trust policies can be described, we shall first define trust templates. A trust template serves as a template for creating new trust instances, specifying values to be bound to parameters upon instantiation.

A trust template is essentially a partially instantiated trust instance. It has a name, a list of parameters, and a pair of truster and subject. Each parameter is defined as either a parameter instance or a variable. Recall that a parameter instance is a concrete value of the type of the parameter.

The basic structure of a trust policy consists of a set of trust uses matching the set of prerequisite trust instances for the new trust instance. The policy may optionally include another set of trust uses for matching trust instances whose existence prevents the creation of the new trust instance. Conditions and rules may be specified to constrain parameters in trust instances and to set values for variables. Additionally, it is possible to associate specific actions and/or validity conditions with new trust instances. This policy states that the policy owner (namely, self) believes T3 regarding principal Z, provided she believes T1 about Z, and Y believes T2 about Z at the same time. The language features a variable matching rule, whereby the value of all occurrences of the same variable must match. Therefore to obtain a T3 instance according to the above policy, valid instances of T1 and T2 with matching parameter instances must be presented.

The trust is a non-monotonic concept, *e.g.* an entity can be believed to be malicious. The framework has the notion of distrust. Without clause is the mechanism in the language to support this notion. It allows negative comments/recommendations to be considered. Electively, it means that the trust instances matched by the trust uses in the without clause must not exist for the trust policy to be evaluated with a positive result, i.e. certain negative trust instances must not exist. A typical use is to implement a blacklist mechanism to prevent distrusted principals causing further damage to others. A real-life example is the Better Business Bureau, which in addition to listing good businesses also often lists bad businesses as a warning for consumers.

The variable matching rule provides a coarse-grained constraining instrument for parameters in trust instances. Fine-grained constraints can be specified through the conditional expression in a where clause. A conditional expression operates on: (1) parameter variables in trust and distrust uses, and (2) environmental variables. An environmental variable is a typed name-value pair, whose value is supplied externally at evaluation. An environment consists of a list of environmental variables.

The syntax for conditional expressions is specific and may be local to every policy specification. The only requirement is that a conditional expression must be side-effect free. Expressions used in this thesis include operators for: arithmetic, comparison, logical connectives, regular expressions, groups and principals.

# CHAPTER 4

# CRYPTOGRAPHY AND ITS APPLICATION IN DISTRIBUTED SYSTEMS SECURITY

## 4.1 PUBLIC-KEY CRYPTOGRAPHY

The Public key encryption can be defined as encrypting the message with the help of public key that can again be decrypted by the private key. It also allows message integrity and confidentiality. In the public key cryptography approach the asymmetric key algorithm (Stallings (2004)) is used in place of symmetric key algorithm. The mathematically related key pair is used to design the asymmetric key algorithm. This involves two types of keys, i.e. a secret private key and a public key. With the help of these keys a digital signature is being created by using the private key which provides the protection and the authenticity of the message which can be verified by using the public key. Moreover, asymmetric key algorithm is used in the different techniques used in the public key cryptography. It is defined as the key used to encrypt a message by using the same key which is used to decrypt the message. There is two cryptographic pair of keys: a public key and a private key. The private key is kept confidential where as the public key may be broadly spread (Stallings (2004)). Primarily and broadly used technology around the world is public key cryptography. Various cryptographic algorithms and cryptosystems are used to achieve the desired goals. Various transport layer security, Pretty Good Privacy (PGP) and Gnu Privacy Guard (GPG) are used as internet standards in this approach (Stallings (2004)).

It may be further being pointed that the public-key cryptography has a fundamental problem of the confidence of the public key. In this system a user has to trust irrespective of the fact that whether it is correct or belongs to some other person or it is replaced by some other entity or replaced by the third party which is malicious. The standard approach to rectify this problem is to use a Public-Key Infrastructure (PKI). The public-key infrastructure is used in one or more third parties who is known as certificate authorities and certify ownership of key pairs. The public-key cryptography is divided into two keys i.e. *Public key* and *Private Key* for which it may be noted that

- it is computationally easy for a party *B* to generate a pair (public key $PU_b$, private key $PR_b$).
- it is computationally easy for a sender A, knowing the public key and the message to be encrypted, M, to generate the corresponding cipher text

$$C = E(PU_b, M) \tag{4.1}$$

- it is computationally easy for the receiver *B* to decrypt the resulting cipher text using the private key to recover the original message:

$$MD(PR_b, C) = D[PR_b, E(PU_b, M)] \tag{4.2}$$

- it is computationally infeasible for an adversary, knowing the public key, $PU_b$, to determine the private key, $PR_b$.
- It is computationally infeasible for an adversary, knowing the public key, *PU* and a cipher text, *C,* to recover the original message, M.

Moreover, although useful but not necessary for all public-key applications that the two keys can be applied in either order:

$$M = D[PU_b, E(PR_b, M)] = DD[PR_b, E(PU_b, M)] \tag{4.3}$$

## 4.2 CRYPTOSYSTEM

A cryptosystem (Kaufman (2002)) can be defined as a tool for encryption and decryption used to implement in different algorithms. There are various cryptosystem used for the purpose of encryption and decryption such as public key cryptosystem, symmetric- key, asymmetric –key and hybrid cryptosystem.

The essential elements of a public-key encryption scheme are explained in *Fig.* 4.1. There is source *A* that produces a message in plaintext, X = [$X_1$, $X_2$, ...$X_m$ ]. The M elements of X are letters in some finite alphabet. The message is intended for destination *B*. *B* generates a related pair of keys: a public key, $PU_b$, and a private key, $PR_b$. $PR_b$ is known only to *B,* whereas $PU_b$ is publicly available and therefore accessible by *A* ensuring message secrecy or confidentiality. In this case, *A* prepares a message for *B* and encrypts it using A's private key before transmitting it. *B* can decrypt the message using A's public key. Because the message was encrypted using A's private key, only *A* could have prepared the message, therefore, the entire encrypted message serves as a *digital signature*. In the preceding scheme, the entire message is encrypted. Each document must be kept in plaintext to be used for practical purposes. A copy also must be stored in cipher text so that the origin and contents can be verified in case of a dispute. So this provides non repudiation. It is important to emphasize that the encryption process depicted in Figure 4.1 does not provide confidentiality. That is, the message being sent is safe from alteration but not from eavesdropping. This is obvious in the case of a signature based on a portion of the message, because the rest of the message is transmitted in the clear. There is no protection of confidentiality because any observer can decrypt the message by using the sender's public key.

Table 4.1 summarizes some of the important aspects of symmetric and public-key encryption. To discriminate between the two, we refer to the key used in symmetric encryption as a secret key. The public key and the private keys are used for asymmetric functioning.

Asymmetric cryptosystem is applied to derive confidentiality and authentication as shown in Figure 4.1.

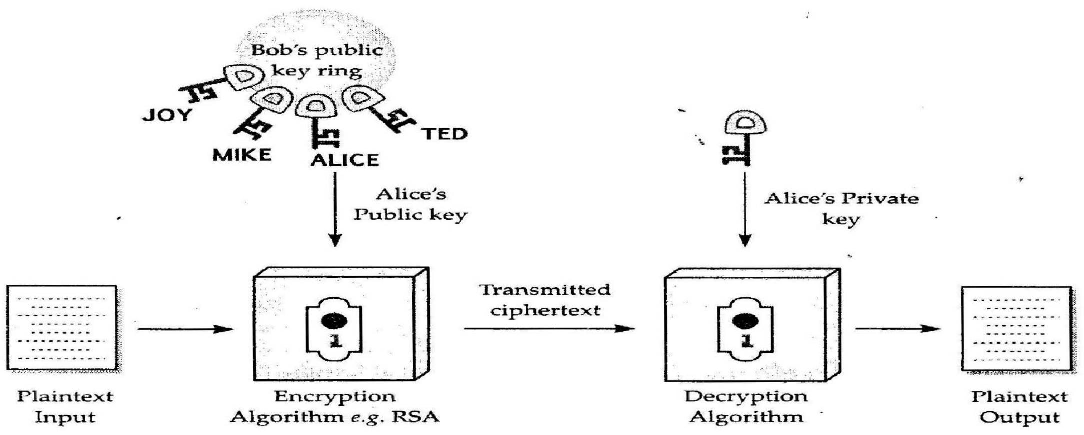
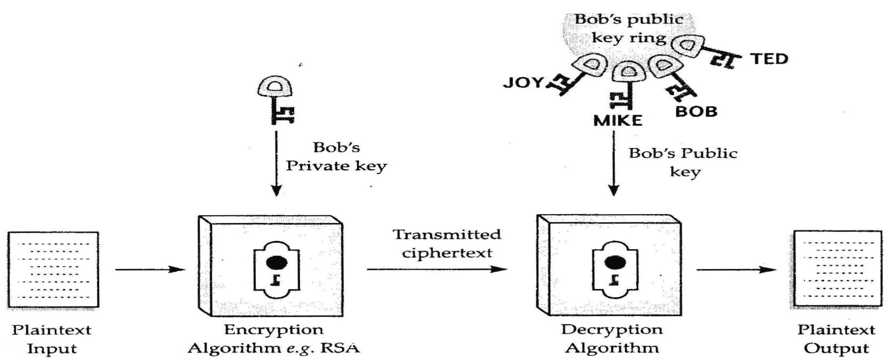


Figure 4.1 Public Key Encryption (Confidentiality) (Kaufman (2002))



Authentication
Figure 4.2 Cryptosystem (Kaufman (2002))

| **Conventional Encryption** | **Public Key Encryption** |
|---|---|
| **Needed to work** | **Needed to work** |
| The same algorithm with the same key is used for encryption and decryption. | One algorithm is used for encryption and decryption with a pair of keys, one for encryption and one for decryption. |
| The sender and receiver must share the algorithm and the key. | The sender and receiver must each have one of the matched pairs of keys (not the same one) |
| **Needed for security** | **Needed for security** |
| The key must be kept secret. | One of the two keys must be kept secret. |
| It must be impossible or at least imprac-tical to decipher a message if no other information is available. | It must be impossible or at least imprac-tical to decipher a message if no other information is available. |
| Knowledge of the algorithm plus samples of cipher text must be insufficient to deter – mine the key. | Knowledge of the algorithm plus one of the keys plus samples of cipher text must be insufficient to determine the other key. |

**TABLE 4.1 CONVENTIONAL AND PUBLIC-KEY ENCRYPTION**

It is, however, possible to provide both the authentication function and confidentiality by a double use of the public-key scheme as given in Equation 4.4.

$$Z = E(PU_b, E(PR_a, X)$$

$$X = D(PU_a, E(PR_b, Z)) \qquad (4.4)$$

In this case, we begin as before by encrypting a message, using the sender's private key. This provides the digital signature. Next, we encrypt again, using the receiver's public key. The final cipher text can be decrypted only by the intended receiver, who alone has the matching private key. Thus, confidentiality is provided. The disadvantage of this approach is that the public-key algorithm, which is complex, must be exercised four times rather than two in each communication.

The combination of public-key cryptosystem and symmetric-key cryptosystem are called as a hybrid cryptosystem (Kaufman (2002)). A hybrid cryptosystem can be developed with the help of two different cryptosystems. K*ey encapsulation* scheme, which is a public-key cryptosystem, and *data encapsulation* scheme, which is a symmetric-key cryptosystem. Public and private keys are same in the hybrid cryptosystem which is used in the public-key system which is similar to the key encapsulation scheme.

### 4.2.1 Public-Key Cryptanalysis

The tool to break the code is called Cryptanalysis and it tells (Kaufman (2002)) *how system works* and *how to find out a secret key*. Cryptanalysis is also used to refer to any attempt to circumvent the security of other types of cryptographic algorithms and protocols. Cryptanalysis usually excludes methods of attack that do not primarily target weaknesses in the actual cryptography. There are four types of cryptanalytic attack:

1. ***Cipher text-only.*** In this type of cryptanalytic attack, the cryptanalyst access only to a collection of code texts.
2. ***Known-plaintext.*** In this type of attack, the attacker has a set of coded text to which he knows to the relevant plaintext.
3. ***Chosen-plaintext.*** In this the attacker can obtain the coded text into an arbitrary set of plaintexts of his own choice.
4. ***Related-key attack.*** This type of attack is like the above attack the only difference is that the attacker can obtain coded text encrypted with the help of two different keys. These two keys are unknown, but have some relationship in between them.

Further, the attacks can be characterized on the basic of the computational resources which are time, memory and data.

### 4.2.2 Applications of Public-key Cryptosystem

With help of two keys *i.e.* private and public key used in cryptographic algorithm the Public-key systems is characterize. There are three types of the usages of public-key cryptosystems: (Kaufman (2002))

1. ***Encryption/decryption.*** The sender encrypts a message with the recipient's public key.
2. ***Digital signature.*** In cryptographic algorithm, the privacy of the message is achieved with the help of the sender signing with its private key and which applied to the message or some part of the message.
3. **Key exchange.** Two sides cooperate to exchange a session key. Several different approaches are possible, involving the private key(s) of one or both parties.

## 4.2 THE RSA ALGORITHM

In cryptography, **RSA** which stands for **Rivest, Shamir and Adleman** who first publicly described it, is an algorithm for public-key cryptography. It is the first algorithm known to be suitable for signing as well as encryption, and was one of the first great advances in public key cryptography. RSA is widely used in electronic commerce protocols (Kaufman (2002)).

The RSA algorithm was publicly described at MIT. The RSA algorithm involves three steps: key generation, encryption and decryption.

### 4.2.1 Key Generation

RSA (Kaufman (2002)) involves a public key and a private key. The public key can be known to everyone and is used for encrypting messages. Messages encrypted with the public key can only be decrypted using the private key. The keys for the RSA algorithm are generated the following way:

1. Choose two distinct prime numbers *p* and *q*.
2. For security purposes, the integer's *p* and *q* should be chosen uniformly at random and should be of similar bit-length. Prime integers can be efficiently found using a primarily test.
3. Compute $n = pq$ (4.5)

   n is used as the modulus for both the public and private keys
4. Compute $(4)(pq) = (p-1)(q-1)$. (Euler's totient function). (4.6)
5. Choose an integer e such that ex d-----1 mod 4) (n) and l< e< (pq), and e and (pq) share no divisors other than 1. e is released as the public key exponent.
6. Determine *d* (using modular arithmetic) which satisfies the congruence relation.

$$d\, e^{-1} \bmod 0\,(n) \tag{4.7}$$

where d is computed using the extended Euclidean algorithm and is kept as the private key exponent.

The public key consists of the modulus *n* and the public exponent *e*. The private key consists of the private exponent *d* which must be kept secret.

**Encryption**

$$C = M^{e} \bmod n \tag{4.8}$$

**Decryption**

$$M = C^{d} \bmod n = (M^{e})^{d} \bmod n = -Me^{d} \bmod n \tag{4.9}$$

**EXAMPLE**

Let $p$ = 61 and $q$ = 53. Make sure that these prime numbers are distinct.

1. Compute $n= p.q.$
2. Compute the totients of product. For primes, the totient is maximal and equals the prime minus one. Therefore ф $(pq) = (p -1) (q -1)$ = 60 x 52 = 3120
3. Choose any number $e$> 1 that is coprime to 3120. Such that, ex $d$ 1 mod ф)(n), e = 17
4. Compute $d$ such that $d\ e^{-1}$ mod (n) by computing the modular multiplicative inverse of $e$ modulo :

   $d$ = 2753

   17.2753 = 46801 and 46801 mod 3120 = 1, this is the correct answer.

   The public key is (n = 3233, $e$ =$17$). For encryption purpose.
   The private key is ($n$= 3233, d = 2753. For decryption purpose.

   if $m$ =123 then we calculate $c$ = 855.

### 4.2.2 Security of RSA

There are four different approaches used in attacking in the RSA algorithm:

1. **Brute force.** It involves all possible secret keys.
2. **Mathematical attacks.** In mathematical attack we are using different techniques, which is similar in effort to factor the product of two primes.
3. **Timing attacks.** Timing attack is dependent on the running time of the decryption algorithm.
4. **Chosen cipher text attacks.** This attack executes the characteristic of the RSA algorithm.

### 4.2.3 Key Management

There are different procedures which support in Key management:

1. Initialization of systems/users within a domain;
2. Generation, distribution and installation of keying material;
3. Controlling the use of keying material;
4. Update, revocation and destruction of keying material; and
5. Storage, backup/recovery and archival of keying material.

### 4.2.4 Distribution of Public Keys

There are four different techniques used for the distribution of public keys, namely Broadcast the key, Publicly available dynamic directory, Public-key authority, Public-key certificates. We define and discuss each type briefly below.

**Broadcast the key.** In this approach (Forouzan (2008)) any participant can send his or her public key to any other participant or broad cast the key to the community at large. This approach is convenient, and has major weakness. Anyone can forge such a public announcement. That is, some user could pretend to be user "X" and send a public key to another participant or broadcast such as a public key. Before the user discovers the forgery and alerts other participants, the forger is able to read all encrypted messages intended for others and can use the forged keys for authentication.

**Publicly available dynamic directory.** It includes the following elements such as the authority that maintains a directory with an entry for each participant; each participant registers a public key with the directory authority. Registration would have to be in person or by some form of secure authenticated communication.

A participant may replace the existing key with a new one at any time, either because of the desire to replace a public key that has already been used for a large amount of data, or because the corresponding private key has been compromised in some way. Periodically, the authority/organization publishes the entire directory or updates to the directory. Participants could also access the directory electronically.

It is more secure than individual public announcements but still has vulnerabilities. If an opponent succeeds in obtaining or computing the private key of the directory authority, the opponent could authoritatively pass out counterfeit public keys and subsequently impersonate any participant and a message sent to the participant.

**Public-key authority.** Stronger security for public-key distribution can be achieved by providing tighter control over the distribution of public keys from the directory. As before, the scenario assumes that a central authority maintains a dynamic directory of public keys of all participants. In addition, each participant reliably knows a public key for the authority, with only the authority knowing the corresponding private key.

1. A sends a time stamped message to the public-key authority containing a request for the current public key of *B*.

2. The authority responds with a message that is encrypted using the authority's private key, $Plc_{ith}$ . Thus, *A* is able to decrypt the message using the authority's public key. Therefore, A is assured that the message originated with the authority. The message includes the following :

- B's public key, $PU_E$, which *A* can use to encrypt messages destined for *B*.

- The original request, to enable *A* to match this response with the corresponding earlier request and to verify that the original request was not altered before reception by the authority.
- The original timestamp, so *A* can determine that this is not an old message from the authority containing a key other than B's current public key.
- A stores B's public key and also uses it to encrypt a message to B containing an identifier of A ($ID_A$) and a nonce ($N_1$), which is used to identify this transaction uniquely.
- *B* retrieves A's public key from the authority in the same manner as A retrieved B's public key. At this point, public keys have been securely delivered to A and *B,* and they may begin their protected exchange. However, two additional steps are desirable.
- B sends a message to A encrypted with $PLI_a$ and containing A's nonce ($N_1$) as well as a new nonce generated by $B(N_2)$ Because only *B* could have decrypted message (3), the presence of $N_1$ in message (7) assures *A* that the correspondent is *B.*
- A returns $N_2$, encrypted using B's public key, to assure B that its correspondent is A.

Thus, a total of seven messages are required. However, the initial four messages need be used infrequently because both *A* and *B* can save the other's public key for future use, a technique known as caching. Periodically, a user should request fresh copies of the public keys of its correspondents to ensure currency.

**Public-key certificates.** The above scenario is more secure but still it has some drawbacks. So we use a new approach which provides more security than the above technique. This technique uses certificates that can be used by participants to exchange keys without contacting a public-key authority, in a way that is as reliable as if the keys were obtained directly from a public-key authority. Certificate consists of a public key plus an identifier of the key owner, with the whole block signed by a trusted third party. The third party is a certificate authority, such as a government agency or a financial institution that is trusted by the user community. A user can present his or her public key to the authority in a secure manner, and obtain a certificate. The user can then publish the certificate. Anyone needed this user's public key can obtain the certificate and verify that it is valid by way of the attached trusted signature. A participant can also convey its key information to another by transmitting its certificate. Other participants can verify that the certificate was created by the authority. We can place the following requirements on this scheme:

- Any participant can read a certificate to determine the name and public key of the certificate's owner.
- Any participant can verify that the certificate originated from the certificate authority and is not counterfeit.
- Only the certificate authority can create and update certificates.
- These requirements are satisfied by the original proposal in Denning added the following additional requirement:
- Any participant can verify the currency of the certificate.

Each participant applies to the certificate authority, supplying a public key and requesting a certificate.

Application must be in person or by some form of secure authenticated communication. For participant *A,* the authority provides a certificate of the form

$$C_A = E(PR_{auth.}[T_1 \| ID_A \| P_u]) \qquad (4.10)$$

where $PR_{auth}$ is the private key used by the authority and $T$ is a timestamp. $A$ may then pass this certificate on to any other participant, who reads and verifies the certificate as follows:

$$D(Pu_{auth}\, C_A) = D(Pu_{auth} E)(PR_{auth},\, [T_1 \| Pu_a]) = [T_1 \| ID_A \| Pu_a) \qquad (4.11)$$

The recipient uses the authority's public key, *Pll*$_{auth}$ to decrypt the certificate. Because the certificate is readable only using the authority's public key, this verifies that the certificate came from the certificate authority. The elements $D_A$ and pie provide the recipient with the name and public key of the certificate's holder. The timestamp *T* validates the currency of the certificate. The timestamp counters the following scenario. A's private key is learned by an adversary. A generates a new private/public key pair and applies to the certificate authority for a new certificate. Meanwhile, the adversary replays the old certificate to *B*. If *B* then encrypts messages using the compromised old public key, the adversary can read those messages.

### 4.2.5 Distribution of Secret Keys Using Public-key Cryptography

Once public keys have been distributed to the communicating parties, public-key encryption provides the distribution of secret key$_e$5.--t̄6 be used for conventional encryption.

### 4.2.6 Simple Secret Key Distribution

If *A* wishes to communicate with *B,* the following procedure is employed:

1. A generates a public/private key pair *PLI„ PR,* and transmits a message to *B* consisting of *PU,* and an identifier of *A,* $ID_A$.
2. Generates a secret key, *K„* and transmits it to A, encrypted with A's public key.
3. A computes *D* $_{(PR,,}$ *E* (*PU„* , $K_s$)) to recover the secret key. Because only *A* can decrypt the message, only A and *B* will know the identity of $K_s$.
4. *A* discards *PIT,* and *PR,* and B discards $Pu_a$.

*A* and *B* can now securely communicate using conventional encryption and the session key $K_s$. At the completion of the exchange, both *A* and *B* discard $K_s$.

## 4.2.7 Man in the Middle Attack

In this attack an interceptor can intercept messages and then either relay the intercepted message or substitute another message.

In above case, if art interceptor, E has control of the intervening communication channel, then *E* can compromise the communication in the following fashion without being detected:

*A* generates a public/private key pair *{*$M_a$*, PR,*) and transmits a message intended for *B* consisting of *PU„* and an identifier of *A*, $ID_A$.

- *E* intercepts the message, creates its own public/private key pair ($Pil_e$*, PR,}* and transmits $Pil_e$ *II* $ID_A$ to *B*.
- *B* generates a secret key, $K_s$, and transmits *E* (*PU,,* $K_s$).
- E intercepts the message, and learns *K* by computing *D* (*PR„ E*(*PU*,, $K_s$)).
- *E* transmits *E* (*PU,* $K_s$) to *A*.

The result is that both A and *B* know *K,* and are unaware that *K,* has also been revealed to E. A and *B* can now exchange messages using $K_s$ E no longer actively interferes with the communications channel but simply eavesdrops. Knowing $K_s$ E can decrypt all messages, and both *A* and *B* are unaware of the problem. Thus, this simple protocol is only useful in an environment where the only threat is eavesdropping.

### 4.2.8 Secret Key Distribution with Confidentiality and Authentication

We begin at a point when it is assumed that A and *B* have exchanged public keys by one of the schemes described earlier in this section. Then the following steps occur:

1. A uses B's public key to encrypt a message to *B* containing an identifier of *A*(/$D_A$ ) and a nonce ($N_1$), which is used to identify this transaction uniquely.
2. *B* sends a message to *A* encrypted with *PU* and containing A's nonce ($N_1$) as well as a new nonce generated by *B* ($N_2$). Because only *B* could have decrypted message (1), the presence of $N_1$ in message (2) assures A that the correspondent is *B*.
3. A returns $N_2$ encrypted using B's public key, to assure *B* that its correspondent is *A*.
4. *A* selects a secret key $K_s$ and sends M = E ($Ni_b$, $^{E}{}_{(PR,,}$ *KO*) to *B*. Encryption of this message with B's public key ensures that only B can read it ; encryption with A's private key ensures that only A could have sent it.
5. B computes D (PU, D ($PR_b$, M)) to recover the secret key.

### 4.2.9 A Hybrid Scheme

This scheme retains the use of a key distribution center (KDC) that shares a secret master key with each user and distributes secret session keys encrypted with the master key. A public key scheme is used to distribute the master keys.

## 4.3 DIFFIE-HELLMAN KEY EXCHANGE ALGORITHM

The Diffie-Hellman key Exchange protocol is also known as exponential key agreement and it was developed by Diffie and Hellman in 1976 and published in the ground-breaking paper "New Directions in Cryptography." The protocol allows two users to exchange a secret key over an insecure medium without any prior secrets.

The protocol has two system parameters *p* and *g*. They are both public and may be used by all the users in a system. Parameter *p* is a prime number and parameter *g* (usually called a generator) is an integer less than *p,* with the following property: for every number *n* between 1 and p-1 inclusive, there is a power *k* of *g* such that $n= g^k$ mod *p.*

1. Suppose A and B want to agree on a shared secret key using the Diffie-Hellman key agreement protocol. They proceed as follows.

First, *A* generates a random private key X. and *B* generates a random private Key $X_{b.}$ Both X, and $X_b$ are drawn from the set of integers. Then they derive their public keys using parameters p and g and their private keys.

2. A's public key $K_a = g^x a$ mod p and
3. B's public key $k_b$ $g^x b$ mod p, Now they can compute secret key,
4. A computes secret key $K_1 = (K_b)\ ^x a$ mod p, and
5. B computes secret key $K_2 = (K_a)$; mod p.

Now these two calculations produce identical results:

$K_1 = (K_b\ )^x a \bmod p$

$= (g^x b \bmod p)^x a \bmod p$

$= ({}_g Xb\ )2C,\ {}_{\text{mod}}\ \mathbf{p}$

$= \mathbf{(}\boldsymbol{g}\ \boldsymbol{Xa}\ Xb\ {}_{\text{mo}}\text{d}\ p$

$=K_2$

$K_1 = K_2$

The Diffie-Hellman key exchange is vulnerable to a man-in-the-middle attack. In this attack, an opponent C intercepts A's public key and sends her own public value to *B.* When *B* transmits his public key, *C* substitutes it with her own and sends it to *A*. *C* and *A* thus agree on one shared key and *C* and *B* agree on another shared key. After this exchange, *C* simply decrypts any messages sent out by *A* or *B,* and then reads and possibly modifies them before re-encrypting with the appropriate key and transmitting them to the other party. This vulnerability is present because Diffie-Hellman key exchange does not authenticate the participants. Possible solutions include the use of digital signatures and other protocol variants.

Suppose that user A wishes to set up a connection with user B and use a secret key to encrypt messages on that connection. User A can generate a one-time private key $X_t$, , calculate $K_a$, and send that to user B. User B responds by generating a private value $X_b$ calculating $K_b$, and sending $K_b$ to user A. Bothusers can now calculate the key. The necessary public values p and g would need to be known ahead of time. Alternatively, user A could pick values for p and g and include those in the first message.

**EXAMPLE 1**

Users A and B use the Diffie-Hellman Key Exchange Technique with a common prime p =71 and a primitive root g =7.

Find

*(i)* If User A has a private key $L_A = 5$, what is A's Public Key $M_A$.

*(ii)* If User B has a private key $L_B = 5$, what is A's Public Key $M_B$.

*(iii)* TA/hat is Shared Secret Key.

**<u>Solution.</u>**

(i) A's Public Key $M_A$

$M_A = g^L A$ mod p

$= (7)^5$ mod 71

= 16087 mod 71

= 51

(ii) B's Public Key

$M_B = g^L B \bmod p$

$= (7)^{12} \bmod 71$

$= 3841287201 \bmod 71$

$=4$

(iii) Shared Secret Key

At user A $\quad K = (M_B)^{LA} \bmod p$

=(4)5 mod 71

= 1024 moct71

=30

Users A and B use the Diffie-Hellman Key Exchange technique with a common prime p = 23 and a primitive root g= 5. Find secret key if $X_a = 6$ and $X_b = 15$.

A's Public Key $\quad K_a = g^x a \bmod p$

$= 5^6 \bmod 23 = 8$

B's Public Key $\quad K_b = g^X b \bmod p$

$= 5^{15} \bmod 23 =19$

*A computes secret* $K= K_b^{Xa}$ *mod* p

$= 19^6 \bmod 23 = 2$

B *computes secret* $K = K_b{}^x b$ *mod* p

$=8^{15}$ *mod 23 = 2*

## 4.4 ELGAMEL ENCRYPTION

Diffie-Hellman key exchange is based on the public–key cryptography used in asymmetric key encryption algorithm in which the Elgamel encryption system of cryptography is used. It was developed by Taher Elgamel in 1985. Elgamel encryption is used in the free GNU Privacy Guard software, 'recent versions of PGP, and other cryptosystems. The Digital Signature Algorithm is an alternative of the Elgamel signature scheme, which is not same as with Elgamel encryption.

Elgamel encryption can be defined over any cyclic group *G.* Its security depends upon the difficulty of a certain problem in *G* related to computing discrete logarithms. Elgamel encryption consists of three parts - key generator, encryption algorithm, decryption algorithm.

**Key Generation.** The key generator works as follows:

- A generates an efficient description of a multiplicative cyclic group G of order *q* with generator *g*.
- A chooses a random x from (0, 1 ... *q —1*)
- A computes $h = g^x$ .
- A publishes *h,* along with the description of *G, q, g,* as her public key *x.* A retains x as her private key which must be kept secret.

**Encryption.** The encryption algorithm works as follows: to encrypt a message m to A under her public key (G, q, g, h)

- B chooses a random y from (0, 1 ... q —11, then calculates $c_1$ = gY
- B calculates the shared secret s = h'. Since a new S' is computed for every message. "S" is also called an ephemeral key.

The steps above can be computed ahead of time.

- B converts his secret message in into an element in' of G.
- B calculates $c_2 = m^i .s$.
- B sends the ciphertext ($c_1$, $c_2$) to Alice.

**Decryption.** The decryption algorithm works as follows: to decrypt a ciphertext ($c_1$, $c_2$) with her private key, x:

- A calculates the shared secret s
- and then computes m' = $c_2 . s^1$ which she then converts back into the plaintext message, 'in'.

The decryption algorithm produces the intended message, since

$$C_2.s^{-1}=m' h^y (g^{xy})^{-1} = m^r g^{ry} g^{xy}= M^1 \qquad (4.12)$$

The Elgamel cryptosystem is usually used in a hybrid cryptosystem i.e., the message itself is encrypted using a symmetric cryptosystem and Elgamel is then used to encrypt the key used for the symmetric cryptosystem. This allows encryption of messages that are longer than the size of the group G.

### 4.4.1 Security

The security of the Elgamel scheme depends on the properties of the underlying group G as well as any padding scheme used on the messages. If the computational Diffie-Hellman assumption holds in the underlying cyclic group G, then the encryption function is one-way. If the decisional Diffie-Hellman assumption (DDH) holds in G, then Elgamel achieves semantic security. Semantic security is not implied by the computational Diffie-Hellman assumption alone. See decisional Diffie-Hellman assumption for a discussion of groups where the assumption is believed to hold.

Elgamel encryption is unconditionally malleable, and therefore is not secure under chosen cipher text attack. For example, given an encryption ($c_1$, $c_2$) of some (possibly unknown) message m, one can easily construct a valid encryption ($c_1$, $2c_2$) of the message 2m. To achieve chosen cipher text security, the scheme must be further modified, or an appropriate padding scheme must be used. Depending on the modification, the DDH assumption may or may not be necessary.

Other schemes related to Elgamel which achieve security against chosen cipher text attacks have also been proposed. The Cramer—Shoup cryptosystem is secure under chosen cipher text attack assuming DDH holds for G. Its proof does not use the random oracle model. Another proposed scheme is DHAES, whose proof requires an assumption that is weaker than the DDH assumption.

### 4.4.2 Efficiency

Elgamel encryption is probabilistic, meaning that a single plaintext can be encrypted to many possible cipher texts, with the consequence that a general Elgamel encryption produces a 2:1 expansion in size from plaintext to cipher text. Encryption under Elgamel requires two exponentiation; however, these exponentiations are independent of the message and can be computed ahead of time if need be. Decryption only/requires one exponentiation:

### 4.4.3 Decryption

The division s by can be avoided by using an alternative method for decryption. To decrypt ($c_1$, $c_2$) a ciphertext with Alice's private key *x,*

- A calculates s' $=C_1^{(a-x)}$
- *S'* is the inverse of s. This is a consequence of Lagrange's theorem, because s.s' $=g^x.gq^{-x} = gq = 1$
- A then computes m' = $c_2$ . s', which she then converts back into the plaintext message m.

## 4.5 DNA CRYPTOGRAPHY

As modern encryption algorithms are broken, the world of information security looks in new directions to protect the data it transmits. The concept of using DNA computing in the fields of cryptography and steganography has been identified as a possible technology that may bring forward a new hope for unbreakable algorithms. Is the fledgling field of DNA computing the next cornerstone in the world of information security or is our time better spent following other paths for our data encryption algorithms of the future? Research has been performed in both cryptographic and steganographic situations with respect to DNA computing. The constraints of its high tech lab requirements and computational limitations combined with the labour intensive extrapolation means, illustrate that the field of DNA computing is far from any kind of efficient use in today's security world. DNA authentication on the other hand, has exhibited great promise with real world examples already surfacing on the marketplace today. The world of encryption appears to be ever shrinking. Several years ago the thought of a 56-bit encryption technology seemed forever safe, but as mankinds' collective computing power and knowledge increases, the safety of the world's encryption methods seems to disappear equally as fast. Mathematicians and physicists attempt to improve on encryption methods while staying within the confines of the technologies available to us. Existing encryption algorithms such as RSA have not yet been compromised but much fear the day may come when even this bastion of encryption will fall. There is hope for new encryption algorithms on the horizon utilizing mathematical principles such as Quantum Theory however the science of our very genetic makeup is also showing promise for the information security world.

The concepts of utilizing DNA computing in the field of data encryption and DNA authentication methods for thwarting the counterfeiting industry are subjects that have been surfacing in the media of late. How realistic are these concepts and is it feasible to see these technologies changing the security marketplace of today?

DNA-based Cryptography which puts an argument forward that the high level computational ability and incredibly compact information storage media of DNA computing has the possibility of DNA based cryptography based on one time pads.

They argue that current practical applications of cryptographic systems based on one-time pads is limited to the confines of conventional electronic media whereas as small amount of DNA can suffice for a huge one time pad for use in public key infrastructure (PKI).

To put this into terms of the common Alice and Bob description of secure data transmission and reception, they are basing their argument of DNA cryptography on Bob providing Alice his public key, and Alice will use it to send an encrypted message to him. The potential eavesdropper, Eve, will have an incredible amount of work to perform to attempt decryption of their transmission than either Alice or Bob.

Public key encryption splits the key up into a public key for encryption and a secret key for decryption. It's not possible to determine the secret key from the public key. Bob generates a pair of keys and tells everyone his public key, while only he knows his secret key. Anyone can use Bob's public key to send him an encrypted message, but only Bob knows the secret key to decrypt it. This scheme allows Alice and Bob to communicate in secret without having to physically meet as in symmetric encryption methods.

Injecting DNA cryptography into the common PKI scenario, the researchers from Duke argue that we have the ability to follow the same inherent pattern of PKI but using the inherent massively parallel computing properties of DNA bonding to perform the encryption and decryption of the public and private keys.

It can easily be argued that DNA computing is just classical computing, albeit highly parallelized; thus with a large enough key, one should be able to thwart any DNA computer that can be built. This puts the idea of this form of DNA computing at great risk in the field of cryptography. As well, the obstacles of utilizing this kind of technology outside of a lab are extremely high.

### 4.5.1 Advantages of DNA Cryptography

The DNA cryptography has a number of advantages we list below some of them.

1. *Speed* – The Speed of a Conventional computer is approximately 100 MIPS (millions of instruction per second). As implemented by Adleman, combining the DNA strands makes the computations much more easily demonstrate, it is much more 100 times faster than the fastest computer. The inherent parallelism of DNA computing was staggering.
2. *Minimal Storage Requirements* – The memory density of DNA is approximately 1 bit per cubic nanometer where in the conservative storage media it requires cubic nanometers to store 1 bit.
3. *Minimal Power Requirements* – In DNA Computation no power is required for the computation. The chemical bonds that are used for building blocks of DNA can be done without any power resource. There is as such no comparison between the conventional computers.

## CHAPTER 5

# TRUST BASED DISTRIBUTED SYSTEM SECURITY APPROACH BASED ON SOFT COMPUTING

Trust among entities of a distributed system supports secure environment. In this chapter, we propose an approach to compute trust values using fuzzy based approach. The main focus of the approach is on the dynamic behavior of trust values. Mutual authentication is ensured among the communicating entities using trust values. These trust values are transmitted securely using cryptography. The trust values have also been used to generate access control policies.

This chapter is divided into 5 sections; section 2 defines the terminology we have used for defining our grid system also with the data structures used for maintaining the trust values. Section 3 defines the fuzzy approach used to calculate the trust value. Then section 4 defines the various security measures used in this framework. Section 5 is conclusion and future scope.

## 5.1 INTRODUCTION

As discussed by Shehab *et al.* (2010), security methodologies in grid systems are an important aspect in the distributed system security research. Security features required in distributed system are identified as: trust, authentication, encryption and access control as explained by Pallickara *et al.* (2007). Trust is used to ensure secure communication in distributed system message passing as discussed in Anderson (2010). It can be used for implementing security methods in grid environment (Omar *et al*. (2009)). A distributed trust model has been developed by Omar *et al* (2009) for ad hoc mobile networks. A trust and access control based model for distributed system has been developed by implemented by Feng *et al.* (2008). Trust and reputation based access control mechanism have been discussed by Marmol and Derez (2009). For wireless communications, a rendezvous node based trust based security is proposed by Cheng *et al.* (2011). A behavioral pattern based security system is proposed by Ukil and Arit (2011) and Uzunov *et al.* (2012). A new access control model is proposed by Aneta (2012) that ensures data security in distributed system. Trust Web Rank based security approach has been proposed by Karchiolo *et al.* (2012). Fuzzy logic can be used for calculating trust. Many models have been proposed to compute trust using fuzzy. Abdul-Rehman and Hailes (2007) have proposed a trust model where trust is defined in terms of direct trust and recommended trust. In our framework we have direct trust values based on direct communication and updated trust value based on direct and feedback trust values. Fixed weighted (Tang and Chen (2003)) and variable weighted (Liao *et al.* (2009)) fuzzy based models have been proposed for evaluation of trust. As evaluation parameters have dynamic significance, hence variable weighted method is used here.

## 5.2 PROPOSED FRAMEWORK

A grid is composed of a number of independent domains. These domains are referred as organizations. Each organization is an autonomous domain with its own administrative and security policies. These are denoted as vector O= $\{O_1, O_2, O_3 ..... O_n\}$, where, $O_i$ is the ith organization of the grid environment. These organizations comprises of entities. An entity is represented as $e_{ij}^{x}$ if where i represents the organization to which entity e belongs, j represents the grade of an entity within the organization x is used as a name for entity to differentiate from different entities. Grade of an entity refers to the level of trust that an organization has over its own entity. Two entities of the same organization can have the same grade.

Each organization has a manager. $M_i$ denotes the manager of organization $O_i$. Grades are allotted by the organization manager, which is responsible for maintaining the trust values of an organization and its entities. Initially $M_i$ assigns same grades to all the entities $e \in O_i$. Gradually, with feedbacks from communicating entities, grade is updated. For any entity e, we have following types of trust values:

(a) Initial Trust Value: The Trust value assigned for it by any entity with which no communication has occurred in the past. They are based on authentication queries which are either 0 or $t_0$. The scope of these values is before completion of any transaction.

(b) Direct Trust Value: The Trust generated after an entity has carried out a job. It is based on, initial trust, job success rate, error rate, turnaround time. Its value ranges from [$t_0$, 1] and the scope of these values is before completion of job.

(c) Reputation: It depends on trust values by other entities. Its value depends on feedbacks after job completion, trust on an organization and grade within organization. Further, for any organization, we have various trust values,

(a) Initial trust value: It is based on the authentication queries from the highest grade entity. In case, all entities are of the same grade, randomly any entity can be selected.

(b) Direct Trust Value: With the competition of job, direct trust values of entities of organization are updated.

(c) Direct Trust for $O_i$ = Aggregate function (direct trust values of all communicating entities of $O_i$).

The description of the data structures are as follows: Direct Trust Matrix.

An entity $e_{ij}^{x}$ has a trust matrix

$$\begin{array}{c|cccc|}
 & e_{kl}^{y} & e_{mn}^{z} & \cdots\cdots & e_{op}^{a} \\
C_1 & .5 & .7 & & .3 \\
C_2 & .2 & .9 & & .6 \\
C_3 & .1 & .1 & & 0 \\
\end{array}$$

where columns represents the communicating entities and rows represents the contexts of communications. Above matrix represents that $e_{ij}^{x}$ has 0.5 trust over $e_{kl}^{y}$ for context $C_1$ and 0.2 for context $C_2$. So, we can say that entity $e_{kl}^{y}$ is better for context $C_1$ in view point of $e_{ij}^{x}$.

Feedback Matrix: this matrix is sent to the manager of the entity's organization. For example, let us assume that $e_{jk}^{y}$ ε $O_j$ communicates with ∃e ε $O_i$ then, based on its communications its will generate feedbacks and send it to $M_j$ . $M_j$ will aggregate all the received feedback matrices and send to $M_i$.

$$\begin{array}{c|ccc|} & e_{ih}^{y} & e_{in}^{z} & e_{ip}^{a} \\ C_1 & .5 & .7 & .3 \\ C_2 & .2 & .9 & .6 \\ C_3 & .1 & .1 & \text{NA} \\ \end{array}$$

So in the above matrix we can see, that column represents all the entities of organization $O_{i.}$ With respect to different contexts, trust values are assigned to the entities with which communication took place. Since there was no communication of $e_{jk}^{y}$ with $e_{ip}^{a}$ in terms of $C_3$ hence it takes value as NA.

Update Trust Matrix: This matrix is generating by updating the trust matrix with the feedbacks received from different entities. So a join operator is used here to get a combined result from both the matrices.

Updated Trust Matrix = Trust Matrix (+) feedback matrix, here (+) represents join operator.

## 5.3 APPLYING FUZZY INFERENCE ENGINE IN TRUST COMPUTATION

It is a variable weight based fuzzy evaluation.

## 5.3.1 Computation of Direct Trust Matrix

Direct Trust depends on following parameters, initial trust, job success rate, error rate, turnaround time. Now we need to assign weights to these parameters. Importance of a parameter depends on the instant of time, the trust value is computed. For example, if two entities are going to compute for the first time, then whole weightage will be given to initial trust value. Whereas after few successful job completions initial trust will have least weightage. Therefore, these weights will be variable. $W_i$ is weight for Evaluation parameter $E_{i,}$ where E= ($E_1$ $E_2$ $E_3$ $E_4$) is the evaluation parameter vector. Value of evaluation is denoted as $u_{1,}$ $u_2$, …. $u_n$ where $u_i$ ε {0, $\mu_m$}. So,

When $E_i$ is best $u_i = \mu_m$

When $E_i$ is worst $\mu_i = 0$

$M_i$ will be a non increasing differential function in $(0,\mu_m)$

$\lambda i\ (u) \leq 0$ Where uε $(0,\mu_m)$

So $W_i = (u_1, u_2 \ldots\ldots u_n) = \dfrac{\lambda i(\mu i)}{\sum_{j=1}^{n} \lambda j(\mu j)}$

## 5.3.2 Computation of Feedback Matrix

Say few entities of $O_j$ communicate with entities of $O_i$. After completion of job, $O_j$ entities will give feedback of $O_i$ entities to $M_j$. On the basis of which feedback matrix will be generated.

Depending upon the grades of $e_j$ entities (which would be variable); then feedback values will be computed.

| | |
|---|---|
| $E = \{f_1, f_{2,} f_3 \ldots\ldots\ldots\}$ | $f_i$ – feedback from the entity. |
| $\mu = \{\mu_1, \mu_{2,} \mu_3 \ldots\ldots\}$ | $\mu_i$ – grade value of ith entity |

So, here weight values α grade values

So $W_i = (u_1, u_2 \ldots\ldots u_n) = \frac{\lambda i(\mu i)}{\sum_{j=1}^{n} \lambda j(\mu j)}$ condition, entities provide feedback for same context. So, for different contest, different matrices can be developed.

### 5.3.3 Computation of Update Matrix

We use the joint operation in the following matrix.

As, feedback will be received from different managers

So $E = [Mf_1, Mf_2, \ldots\ldots\ldots Mf_{12}]$ where $Mf_i$ is feedback from $M_i$

$\mu = (\mu_1, \mu_2 \ldots\ldots\ldots \mu_n)$

μ = trust value of any organization.

So, here weights α Organization trust value.

So, $W_i = (u_1, u_2 \ldots\ldots u_n) = \frac{\lambda i(\mu i)}{\sum_{j=1}^{n} \lambda j(\mu j)}$

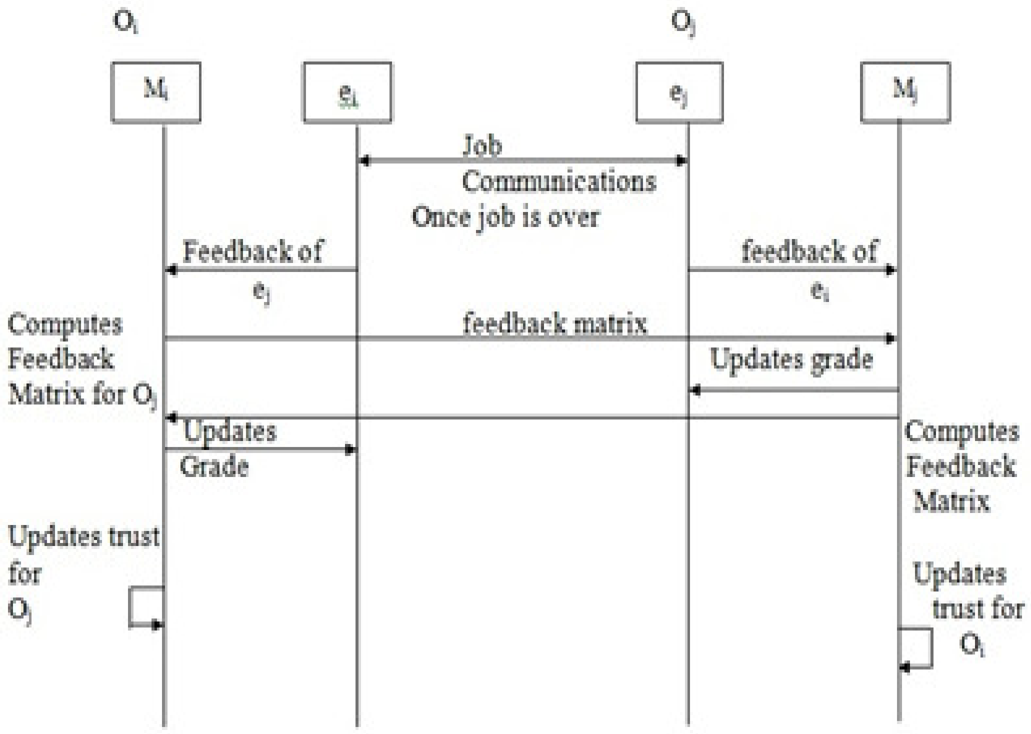


Figure 5.1 Space Time Diagram Values for Trust and Grade Values

## 5.4 DEFENSE IN DEPTH SECURITY

Whenever a new entity/organization wishes to be the part of grid, authentication is required. Security threat could be both ways, it may be possible that a hacker is trying to be a part of grid or, it may be possible that a legitimate user is getting connected to a spoofed grid component, so Mutual Authentication is required. In our framework authentication takes place at two levels. Time line diagram below shows how authentication is carried out.

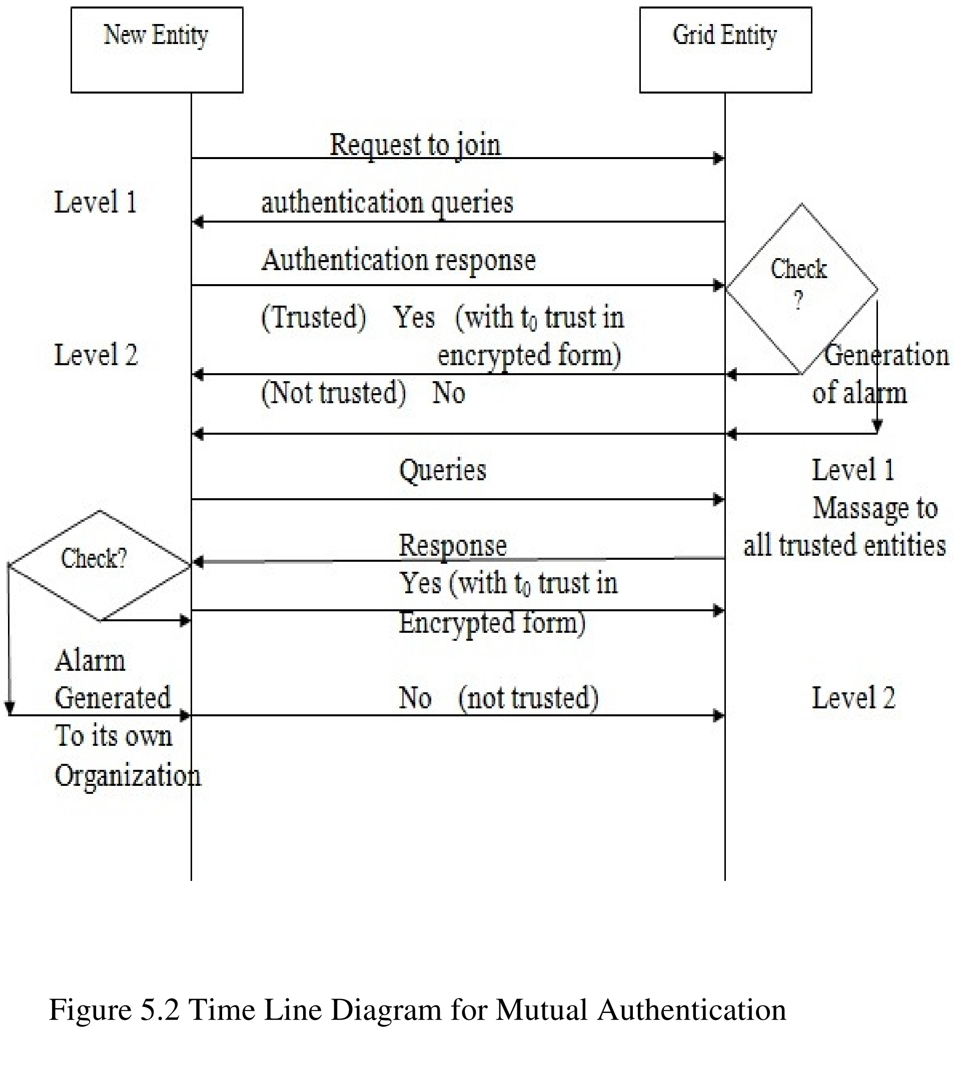


Figure 5.2 Time Line Diagram for Mutual Authentication

Level 1: Whenever a new entity wishes to join a grid, it will send request to particular grid entity to which it wants to communicate. This grid entity will check for authentication, by generating certain queries. Only legitimate user will have the correct answers to those queries. In case correct responses are received, grid entity will sent initial trust value ($t_o$).Else if wrong response will be there then grid entity will assign trust = 0, and an alarm will be generated for all grid entities informing about this unidentified entity.

Level 2: When grid entity sends the initial trust, the trust value is sent using public key cryptography. Therefore, an entity can have access to the trust value, only if it has the key to it. Suppose a hacker in any way gets the responses of initial queries, he will be able to cross the first level authentication, but cannot cross level 2 unless presence of key.

Keys should be changed periodically because in grid computing, entities may join and leave the grid dynamically, even during the execution. So, Stateless Key Management can be used to securely maintain the keys.

The new entity joining the grid must also be sure of the authentication of grid. Therefore, mutual authentication is necessary. New entity will also generate queries to which grid responses are checked and accordingly trust is established. Here also two levels of security are maintained.

### 5.4.1 Access Control

Initially, a new entity will have limited access to resources corresponding to the initial trust $t_0$. As the trust values of a new member increases, he is allowed to get more access to resources. Thus, access control policies vary according to the trust values. The manager's of each entity are responsible for maintaining the access control. Managers maintain the trust value of each entity of other organizations, on the basis of feedback from its own entities.

Say, $e_{ij}^{x}$ want to communicate to $e_{kl}^{y}$ then $M_k$ is responsible for access control. $M_k$ checks the trust value from feedback matrices and accordingly allows $e_{ij}^{x}$

## 5.5 TRUST VALUES

Updated Trust Values (UTV) is calculated using two parameters; Trust Value (TV) and Feedback (FB).

The block diagram of proposed system is as follows:

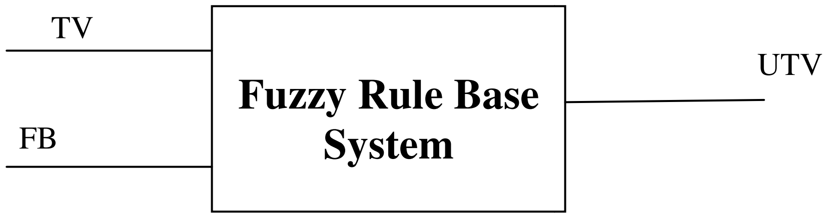


Figure 5.3 Block Diagram of Proposed System

The Membership function parameters are as follows:

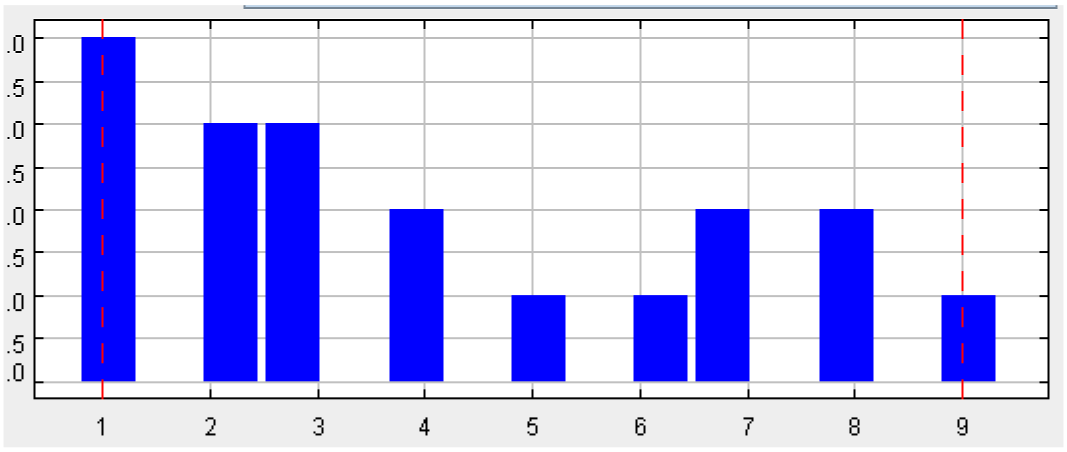


Figure-5.4 Data function of TV

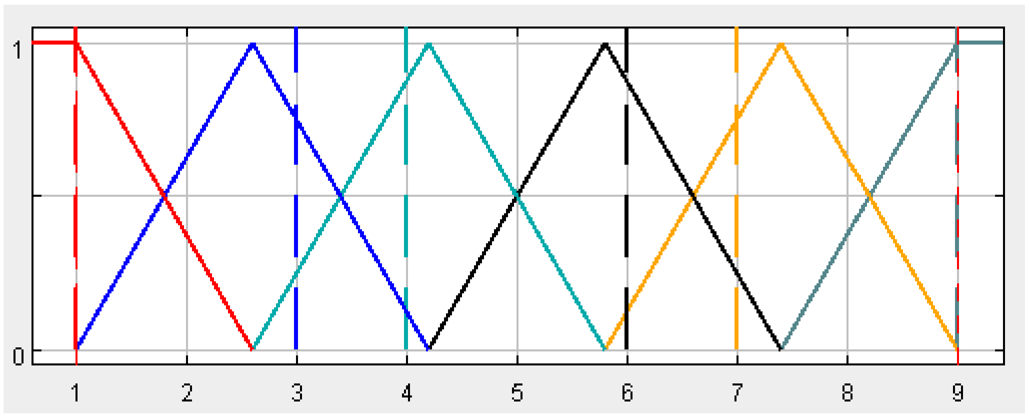


Figure-5.5 Member function of TV

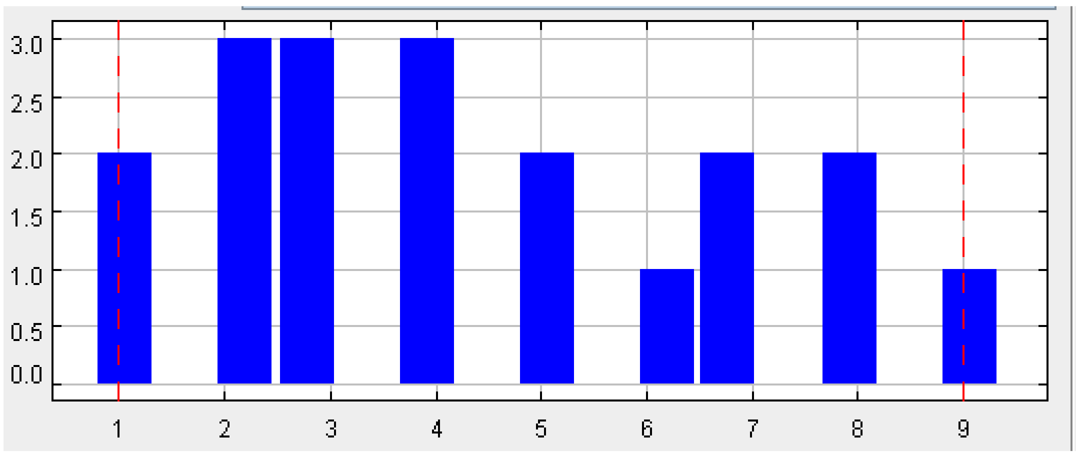


Figure-5.6 Data function of FB

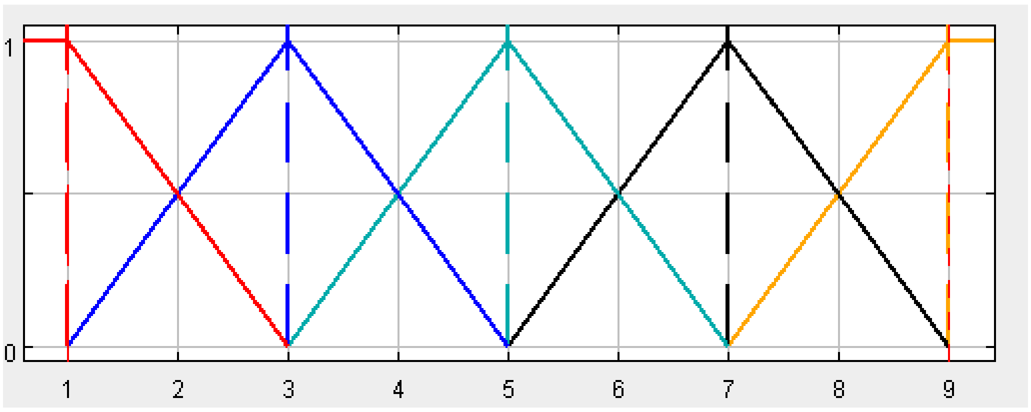


Figure-5.7 Member function of FB

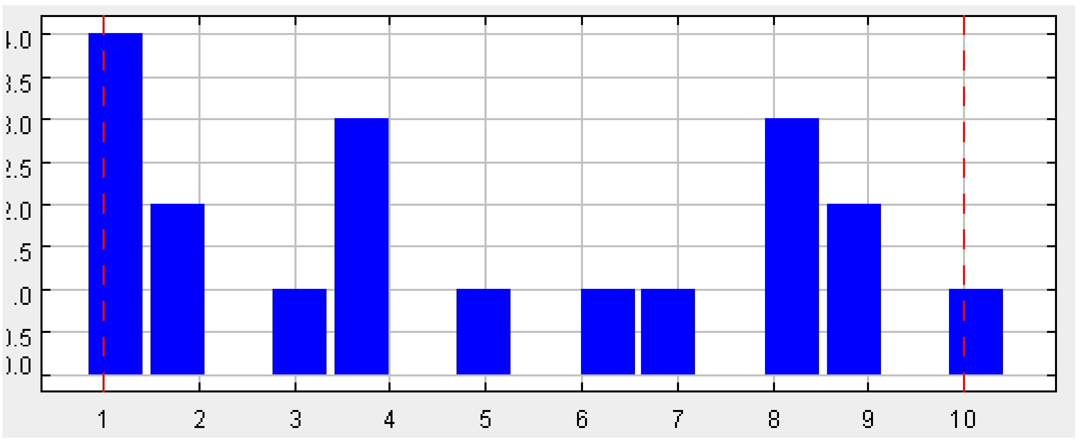


Figure-5.8 Data function of UTV

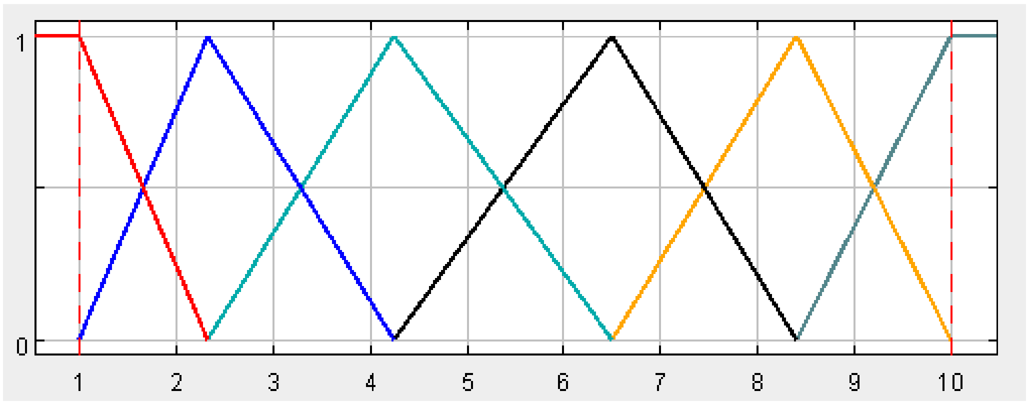
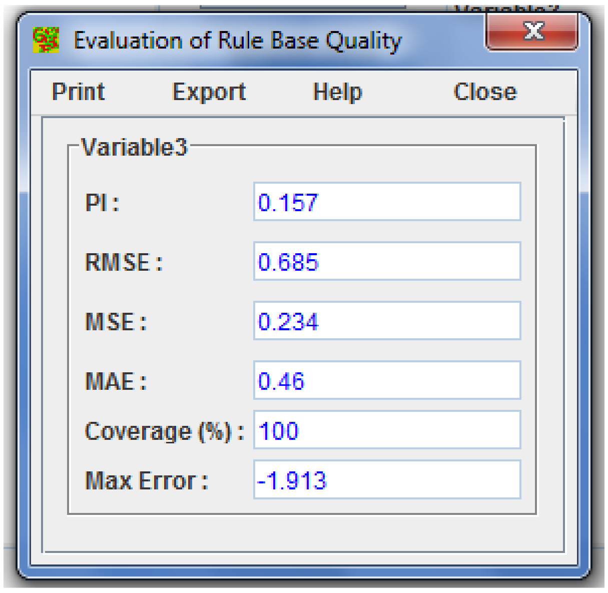


Figure-5.9 Member function of UTV

The Rule Base is as follows:



Variable3

| | |
|---|---|
| PI : | 0.157 |
| RMSE : | 0.685 |
| MSE : | 0.234 |
| MAE : | 0.46 |
| Coverage (%) : | 100 |
| Max Error : | -1.913 |

Figure 5.10 Rule Based Diagram

| TV | FB | UTV |
|---|---|---|
| Low | Low | Low |
| Low | Avg | Low |
| Avg | Low | Avg |
| Avg | High | Avg |
| High | Avg | High |
| High | High | High |
| High | Low | Avg |
| Low | High | Avg |
| Avg | Low | Low |

The inference engine functioning and parameters calculation are as follows:

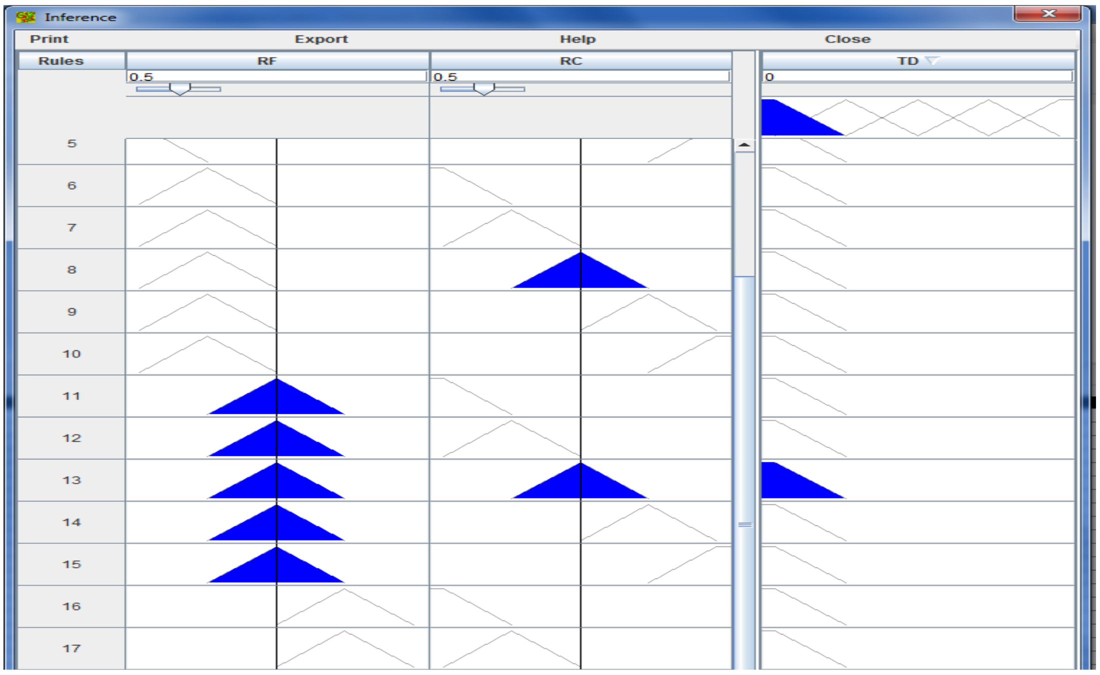
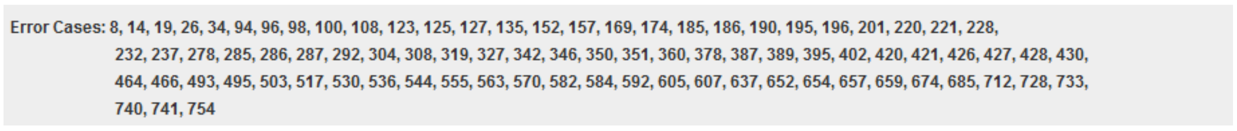
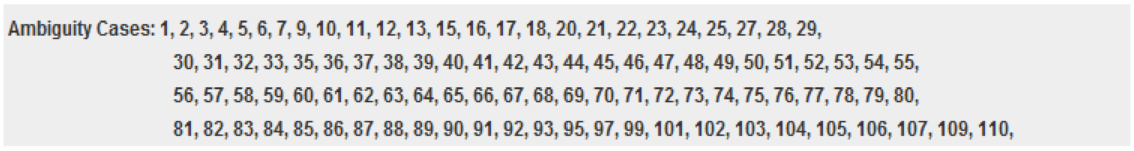


Figure 5.11 Inference engine functioning

The Error cases are as follows:


```
Error Cases: 8, 14, 19, 26, 34, 94, 96, 98, 100, 108, 123, 125, 127, 135, 152, 157, 169, 174, 185, 186, 190, 195, 196, 201, 220, 221, 228,
             232, 237, 278, 285, 286, 287, 292, 304, 308, 319, 327, 342, 346, 350, 351, 360, 378, 387, 389, 395, 402, 420, 421, 426, 427, 428, 430,
             464, 466, 493, 495, 503, 517, 530, 536, 544, 555, 563, 570, 582, 584, 592, 605, 607, 637, 652, 654, 657, 659, 674, 685, 712, 728, 733,
             740, 741, 754
```


```
Ambiguity Cases: 1, 2, 3, 4, 5, 6, 7, 9, 10, 11, 12, 13, 15, 16, 17, 18, 20, 21, 22, 23, 24, 25, 27, 28, 29,
                 30, 31, 32, 33, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55,
                 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80,
                 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 95, 97, 99, 101, 102, 103, 104, 105, 106, 107, 109, 110,
```

## 5.6 CONCLUSION

Trust values have been used to carry out authentication and access control, ensuring a secure grid environment. Fuzzy logic concept is utilized for evaluating the trust. In future, we propose to work on decreasing the computations required for generating and updating trust values. Also reputation parameter can be added to ensure secure communication.

# CHAPTER 6

# ENSURING DISTRIBUTED SYSTEM SECURITY USING DNA CRYPTOGRAPHY AND TRUST BASED METHOD

Distributed system security is well implemented by trust management systems. There is no need for resolving identities in the authorization decisions. These methods are well expressing the constraints and privileges. A new trust based security framework extended with cryptography approaches is implemented in this chapter.

## 6.1 INTRODUCTION

The applicability of distributed systems has been increased due to the advent of Internet. Multiple resources are designated for this type of computing environment; i.e. CPU cycles, I/O bandwidth and memory. The security approaches are introduced to deal with all the resources available in the computing environment. Main properties of the distributed system are; concurrency of components, lack of global clock and independent failure of components. Except these issues, some other major security issues (Shaheb *et al.* (2010), Xiaoyong *et al.* (2011)) includes; multiple autonomous components, not sharing of components by all users, several points of control and several points of failure.

Different components in distributed systems are sub-divided into other sub-components. The components are given with the multiple interfaces enabling them to communicate with each other. This system runs with multiple processes and these processes are extended on multiple processes. Local Area Network and Intranet, Automated Teller Machine, Network, Internet/World Wide Web and mobile and ubiquitous computing are different examples of distributed systems. Clients send request to the servers to access the data. A security mechanism is needed to hide the content of original message that are directly related to security and privacy. An authentication approach for identification of remote user has also been developed. The detail of service attack and mobile code security are the new challenges in the distributed system security. The term 'trust' proposed by Li and Singhal (2007) can be explained as a requirement for making decisions on communication with other entities. The approximation of trust is an important research line. The value of trust on which the system may allow the interaction is another important research issue. In majority, the trust management is divided into two types, namely r*ule base system* and *Reputation System*. In the rule based systems, the trust is maintained as the role that entity plays.

## 6.2 TRUST MANAGEMENT

In a distributed trust management system (Marmol and Perez (2009), Schryen *et al.* (2011), Marmol and Perez (2010), Aikebaier *et al.* (2011), Zhong *et al.* (2009), Maiden *et al.* (2011)); 'trust' is represented by rules, the rights are granted for the other system based on the rules in a user requesting system.

| | |
|---|---|
| Specification | Naming delegation policies |
| Implementation | Chain discovery certificates |
| Applications | PGP-PKI, AC, *etc*. |

**TABLE 6.1 Rule Based Distributed Trust Management Systems**

This is tried to capture the psychological notion of trust in a reputation based trust management system. All the passed interactions are playing a big role in making a trust decisions. In a reputation system, evaluated participants are giving interaction and feedback. The positive feedbacks provide enhancement in the reputation. The collective experience of all participants expresses the reputation of whole system.

| **S. No.** | **Name of Project** | **Description** | **Reference** |
|---|---|---|---|
| 1. | Policy Maker AWK | First example of trust management engine which processes the signed request which are embodied in the trust management system. | Blaze *et al* (1996) |
| 2. | Key Note | Implemented for carrying out experimental work on Policy Maker. | Blaze (2014) |
| 3. | REFEREE | Credentials are used for designing which directly authorize actions in place of subdividing the authorization task in the authentication and access control mechanisms. | Chu *et al* (1997) |
| 4. | Simple Public Key | All programming of assertions like polices and credentials are | |

| | | | |
|---|---|---|---|
| | Infrastructure (SPKI) | explained. This is standard format of authorization certificates. | Ellison *et al* (2014) |

**TABLE 6.2 List of ongoing trust based projects**

## 6.3 PROPOSED APPROACH

A model for distributed system security framework has been proposed. This is considered as a rule based approach for quantifying the trust values that is further post – processed with the help of the reputation approach.

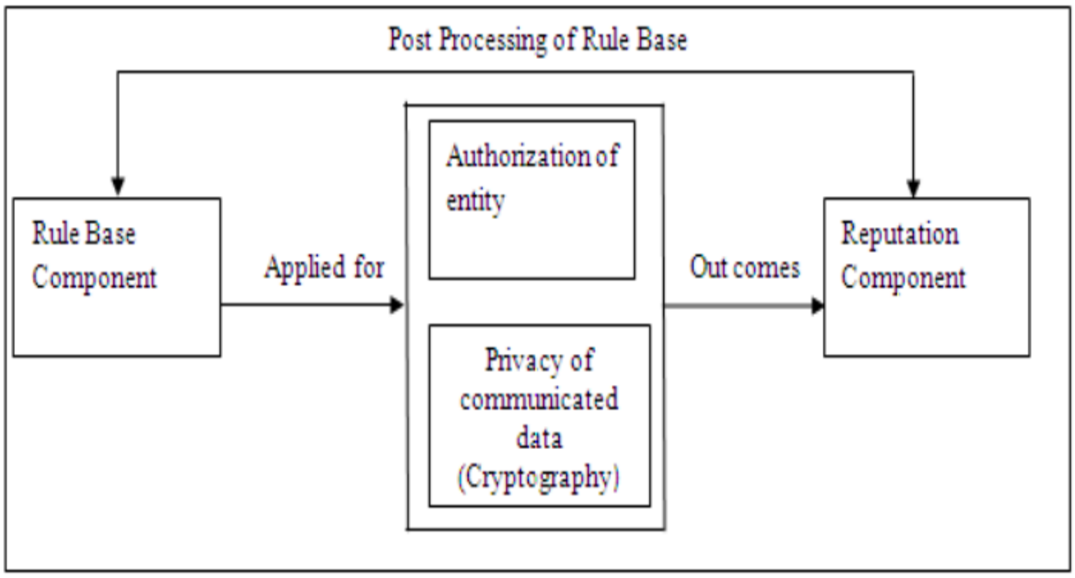


Figure 6.1 Proposed Security Framework Based on Trust

Trust degree is the value obtained by the entity for the existing rule base as expressed in this framework. This is represented by 3 tuples {EID, TD, RF}.

Here, EID = Entity ID, TD = Trust Degree and RF = Reputation Factor.

At initial level, the TD is calculated by the rule base which is updated by post processing using the following transfer function.

$$F: TD \longrightarrow X\ RF \qquad TD$$

Initially RF is equal to 1 and it ranges from 0 to 1 based on the calculation of reputation based approach.

## 6.4 REPUTATION COMPONENT

The reputation component includes the following phases; proof collection, reputation factor approximation, reputation confidence as discussed below.

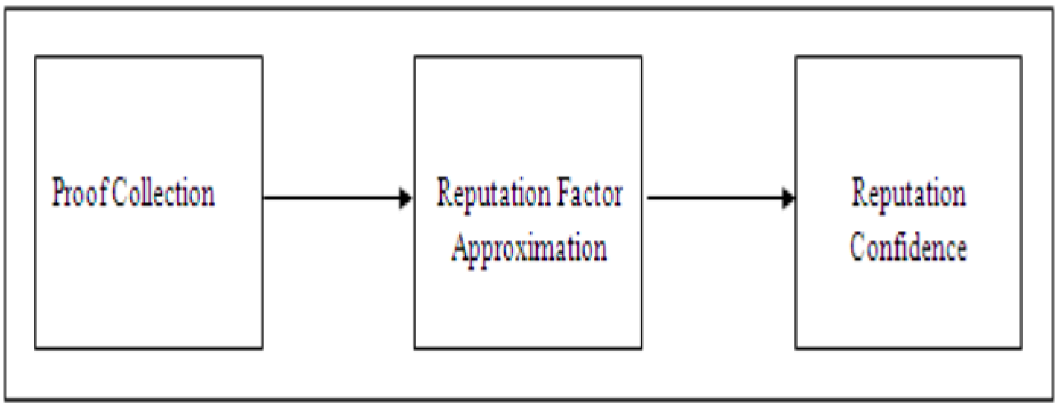


**Figure 6.2 Reputation Component**

Two tuples (RFV, RC) (Figure 6.2) are used for representing Reputation factor (RF) where RFV is the Reputation Factor Value ranging from 0 to 1 & RC is represented for reputation confidence showing the authenticity of RF.

## 6.5 RULE BASED APPROACH

DNA based cryptography approach has been developed by Gehari *et al.* (2004). The approach is as follows:

The encoding of send message is carried out into binary. Now a random sequence is produced. 4 point mutation is carried out at binary plain text, now 8 point crossover operation is performed on the mutated B-Plain Text. The crossover mutated binary string is the decrypted to cipher text. In the decryption method, Decrossover and Demutation operation are carried out using crossover key and mutation key. Decrossover is the reverse of crossover operation and demutation is the reverse of the mutation operation.

## 6.6 PROCEDURE FOR ENCRYPTION

The encryption approach is discussed in *Fig.* 6.3. In Encryption procedure the Plain text is converted into the B-Plain text with help of Binary converse, then this B-Plain text generates the random sequence, on this random sequence we apply the 4 point mutation on plain text with the help of mutation key after this mutation we apply the 8 point crossover between B-plain text and random sequence with the help of crossover key and after this we get finally the encoded text which is called as cipher text.

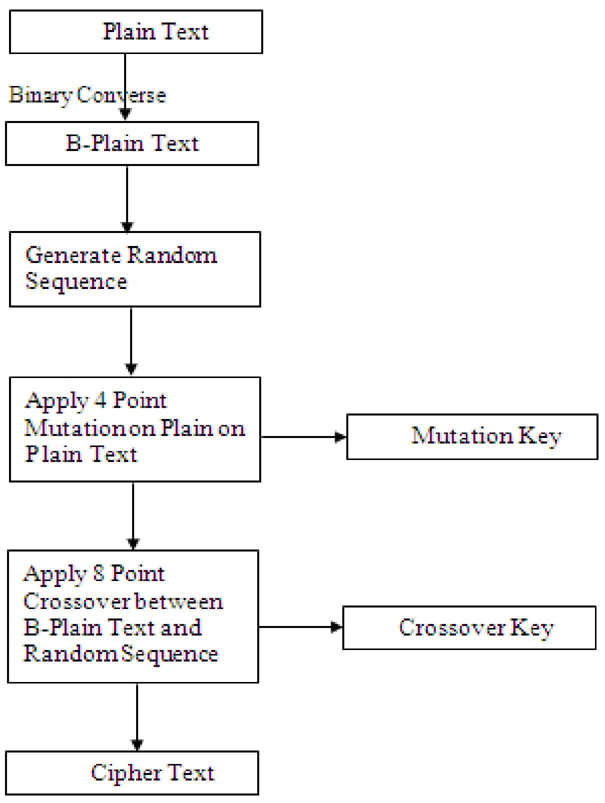

Plain Text
Binary Converse
B-Plain Text
Generate Random Sequence
Apply 4 Point Mutation on Plain on Plain Text
Mutation Key
Apply 8 Point Crossover between B-Plain Text and Random Sequence
Crossover Key
Cipher Text

Figure 6.3 Encryption Operation

## 6.7 PROCEDURE FOR DECRYPTION

The Procedure for Decryption is illustrated in *Fig.* 6.4, in which the cipher text is converted into the decrossover with help of crossover key and after this with the help of mutation key is converted into demutation key, and after this cipher text is converted into the B-plain text and finally it converted into the Plain text i.e. the original text.

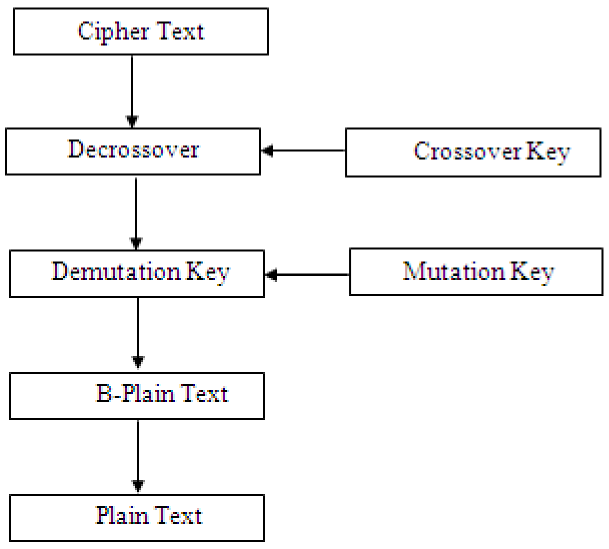


Figure 6.4 Decryption Operation

## 6.8 EXPERIMENTS & RESULTS

The calculation of Reputation Factor has been carried out using two important parameters, Reputation Factor and Reputation Confidence as discussed in section 6.4. the Mamdani FRBS has been used to implement and simulate an fuzzy system. The Block diagram is as follows:

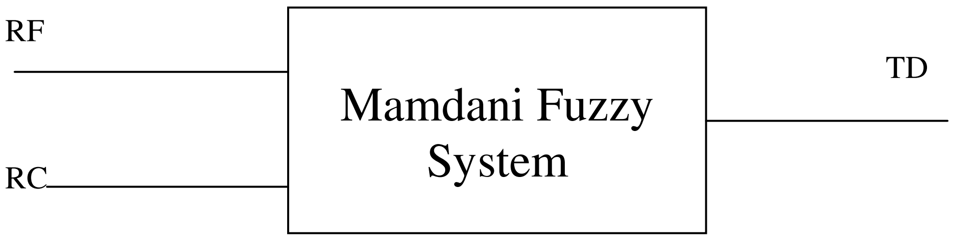


Figure 6.5 Mamdani fuzzy System

The membership functions of these input and output parameters are as follows:

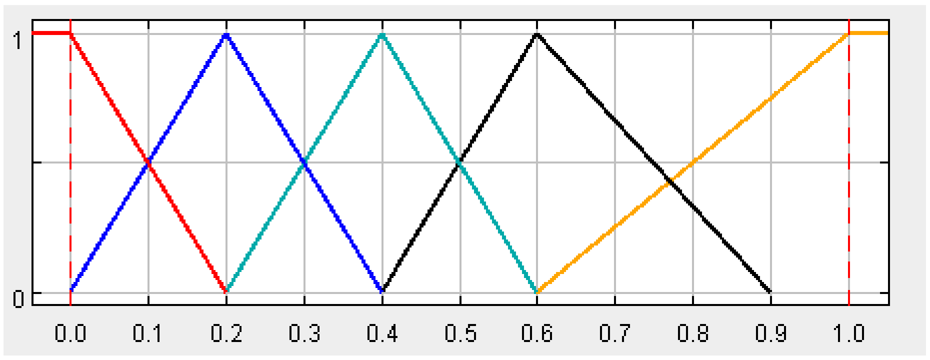


Figure-6.6 Member function of RF

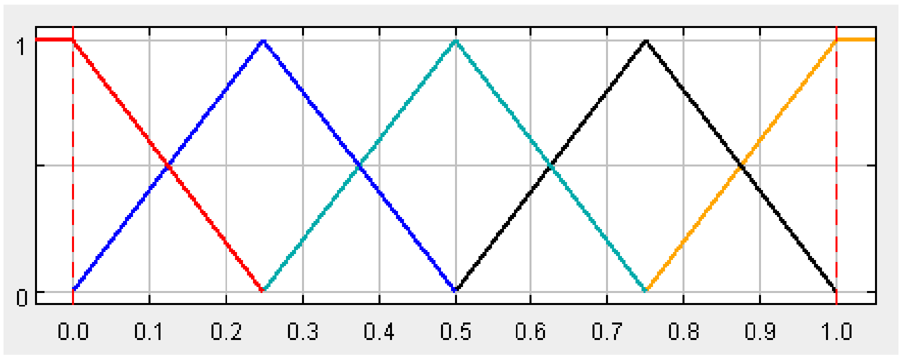


Figure-6.7 Member function of RC

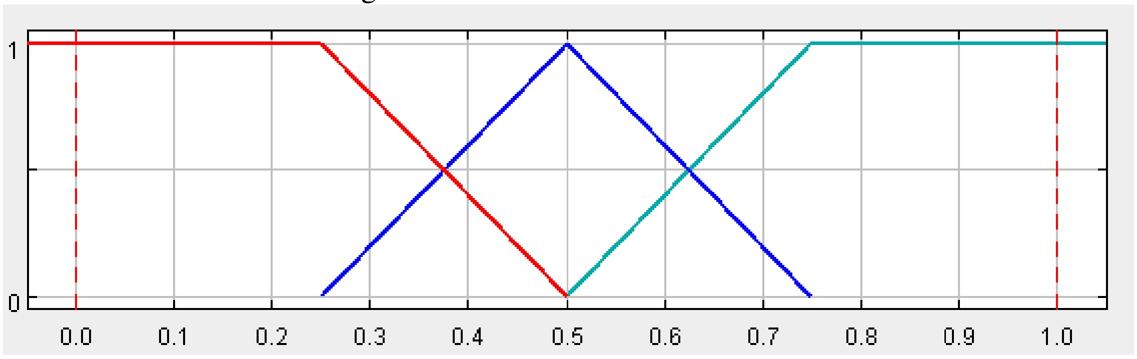
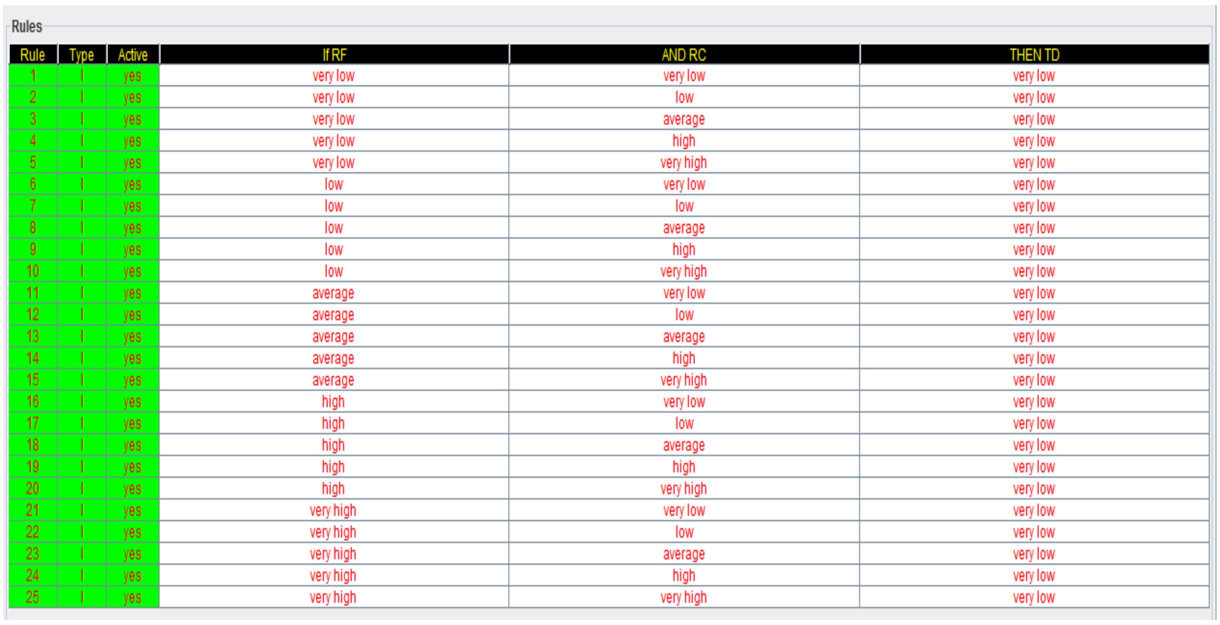


Figure-6.8 Member function of TD

The following Figure 6.9 shows the generated rule base.


Rules

| Rule | Type | Active | If RF | AND RC | THEN TD |
|---|---|---|---|---|---|
| 1 | I | yes | very low | very low | very low |
| 2 | I | yes | very low | low | very low |
| 3 | I | yes | very low | average | very low |
| 4 | I | yes | very low | high | very low |
| 5 | I | yes | very low | very high | very low |
| 6 | I | yes | low | very low | very low |
| 7 | I | yes | low | low | very low |
| 8 | I | yes | low | average | very low |
| 9 | I | yes | low | high | very low |
| 10 | I | yes | low | very high | very low |
| 11 | I | yes | average | very low | very low |
| 12 | I | yes | average | low | very low |
| 13 | I | yes | average | average | very low |
| 14 | I | yes | average | high | very low |
| 15 | I | yes | average | very high | very low |
| 16 | I | yes | high | very low | very low |
| 17 | I | yes | high | low | very low |
| 18 | I | yes | high | average | very low |
| 19 | I | yes | high | high | very low |
| 20 | I | yes | high | very high | very low |
| 21 | I | yes | very high | very low | very low |
| 22 | I | yes | very high | low | very low |
| 23 | I | yes | very high | average | very low |
| 24 | I | yes | very high | high | very low |
| 25 | I | yes | very high | very high | very low |

Figure 6.9 Generated rule base

The procedure of inference engine values and text cases of accusing are as follows:

Wang Mendel Method has been used for the generation of rules which are further used in Rule Base for the purpose of inference engine settings.

Three kinds of approaches are used to carry out the experimentation. Approach 1 deals with the trust based system only, approach 2 deals with cryptographic approach; approach 3 is the implementation of proposed approach. The experiment carries out the analysis of malicious node and trust value assessment. The results are explained below.

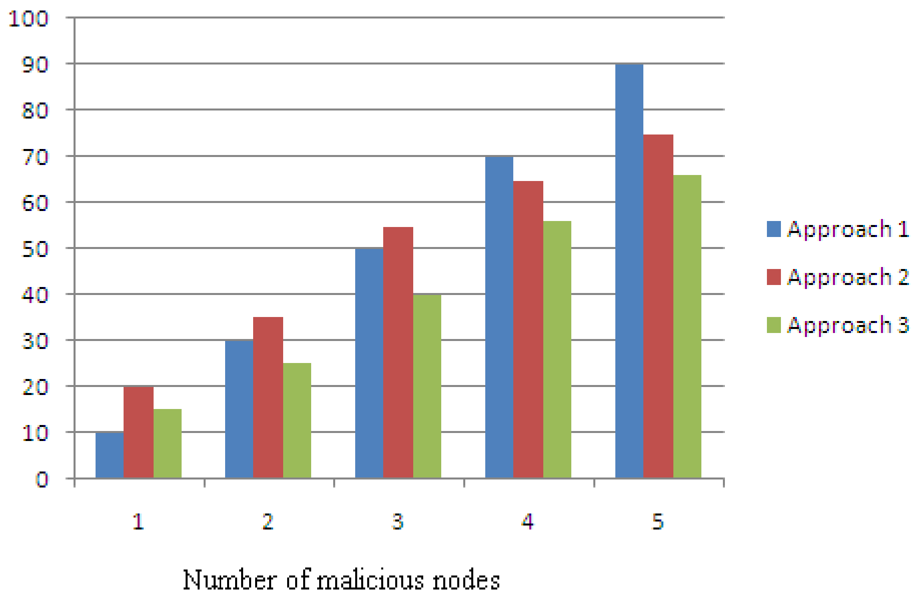


Figure 6.10 Malicious Nodes Analysis

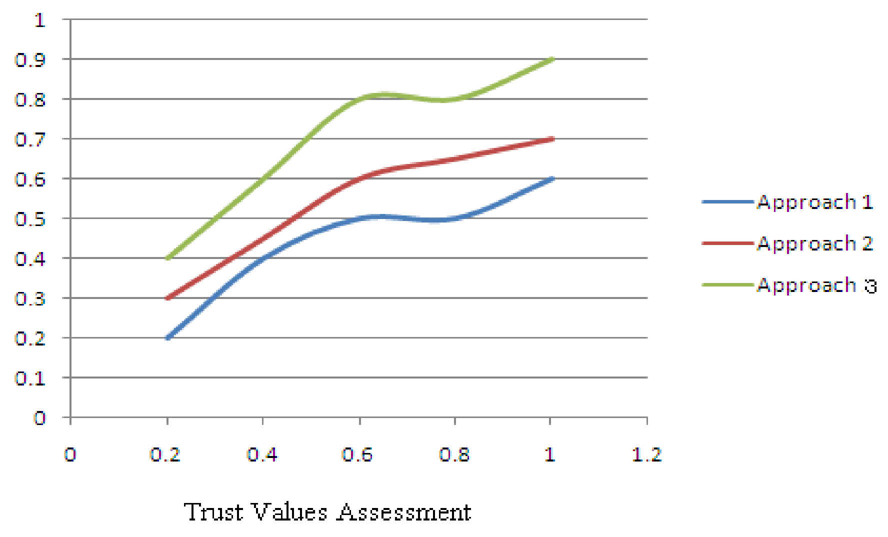


Figure 6.11 Trust Value Analysis

## 6.9 CONCLUSION

To deal with high risk security attacks in distributed systems trust based distributed systems are extremely applicable. Integration of cryptographic methods is an excellent effort towards the development of extremely secure distributed systems. The DNA based cryptography method is integrated with a rule based post processing method dealing with security attacks in the distributed systems.

# CHAPTER 7

# CONCLUSION AND FUTURE SCOPE

Implementing robust security and privacy strategies in distributed systems is a challenging issue. This research work makes a step forward in solving this issue by proposing a generic framework for multi-factor authentication to protect services and resources for unauthorized use. The authentication would be based on trust based systems. Proposed framework would not only demonstrate how to obtain secure multi-factor authentication, but also would be addressing several prominent issues bio-inspired computing applications. To deal with different uncertainties in the proposed model fuzzy logic has been used. The rule base systems are implemented using fuzzy knowledge base approach.

The main characteristics of the proposed approaches are

- Identification of basic security parameters.
- Definition and implementation of security parameters and their inter-relationships.
- Identification of bio-inspired security issues
- Integration of trust based model.

Trust values have been used to perform authentication and access control, ensuring a secure grid environment. Fuzzy logic concept is utilized for evaluating the trust. In future, authors would be working for decreasing the computations required for generating and updating trust values. Also reputation parameter can be added to ensure secure communication.

Having dealt with high risk security attacks in distributed systems trust based distributed systems are extremely applicable. Integration of cryptographic methods is an excellent effort towards the development of extremely secure distributed systems. The DNA based cryptography method is integrated with a rule based post processing method dealing with security attacks in the distributed systems.

I would like to conclude the present thesis with proposal that the work in the thesis can be extended by applying robust bio-inspired mechanisms and computational intelligence approaches.

# REFERENCES

1. Abdul-Rahman, A. and Hailes, S., 2007. Supporting Trust in Virtual Communities, 33rd Hawaii International Conference on System Sciences, Vol. 6, pp. 6007.

2. Aikebaier, A., Enokido, T. and Takizawa, M., 2011. Trustworthy group making algorithm in distributed systems, Human Centric Computing and information Sciences, pp. 1-6, Nov.

3. Anderson, R., 2010. Security engineering: a guide to building dependable distributed systems, Wiley.

4. Aneta, P.M., 2012. Implementation of Access Control Model for Distributed Information Systems Using Usage Control, Security and Intelligent Information System, Vol. 7053, pp. 54-67.

5. Bai, Y., 2008. On distributed system security, International Conference on Security Technology.

6. Bierman, E. and Cloete, E., 2002. Classification of malicious host threats in mobile agent computing. Proceedings of the 2002 annual research conference of the South African institute of computer scientists and information technologists on Enablement through technology, pages 141 -148, 2002.

7. BLAZE, M., Feiginbaum, J. and Lavy, J., 1996. Decentralized Trust Management. In Proc. Of 17th symposium security and Privacy, 164-173, IEEE computer society Press, Loss Alamitos.

8. BLAZE, M., Feigembaum, J., Isonnidis, J. and Keromyties, A. The keynote trust management system, http;//www.cis.upenn.edu/ angelos/keynote.html.

9. Blaze M, Feigonbaum. J., Isoannidis J. and Keromyties, A.D., 1999. The role of trust management in distributed system security in secure internet programming: Security issues for mobile and distributed objects, Vitek and Nensen, Editors,Springer-Verlag, http://www.Dgpter.com/papers/networksec.pdf.

10. Bohossian, V., Fan, C. C., Lemahieu, P. S., Riedel, N. D., Xu, L. and Bnick, J., 2001 Computing in the RAIN : a reliable array of independent nodes, IEEE Transactions on Parallel and Distributed Systems, Vol. 12, Issue 2, pp. 99-114.

11. Bovoselov, A.V., Ansiperov, V. E. and Nikitov, A. A., 2007. Information protection in distributed systems with the help of different layer protocols, Journal of Communications Technology and Electronics, Vol. 52, Issue 10, pp. 1133-1136.

12. Bykoyy, P., Pigovsky, Y., Kochan, V., Sachenko, A., Morkowsy, G. and Aksoy, S., 2008. Genetic algorithm implementation for distributed security systems optimization, 2008 IEEE International Conference on Computational Intelligence for Measurement Systems and Applications.

13. Cappello, F., Ojilali, S., Fedak, G., Herault, T., Magniette, Nen, F., U., and Lodygensky, O., 2005. Computing on large-scale distributed systems: Xtream web architecture, programming models, security, tests and convergence with grid, p2p computing and interaction with grid, 21(3), pp. 417-437.

14. Carchiolo, V., Longheu A., Malgeri M. and Mangioni. G., 2012. Trust assessment: a personalized, distributed, and secure approach, Concurrency and Computation: Practice and Experience, Vol. 24(6), pp. 605-617.

15. Chadwick, D., Oterko, A. and Ball, E., 2003. Role base access control with X.509 attribute certificates, IEEE Internet Computing, 7(2), pp. 62-69.

16. Chang, K.-A., Lee, B.-R. and Kim, T.-Y., 2002. Open authentication model supporting electronic commerce in distributed computing electronic commerce research, Vol. 2, Issue 1-2, pp. 135-149.

17. Chang-Ji, W., Jian-Ping, W. and Hai-Xin, D., 2003. Using attribute certificate to design role- based access control, 4th International Conference on Parallel and Distributed Computing, Applications and Technologies.

18. Cheng, N., Govindan, K. and Mahopatra, P., 2011. Rendezvous based trust propagation to enhance distributed network security, International Journal of Security and Networks Vol. 6(2-3), pp. 112-122.

19. Chu, Y.-H., Feiginbaum, J., Lamacchia, B., Beshick, P. and Straues M., 1997, REFEREE: Trust management for web applications, World Wide Web Journal, 2, pp. 127-139.

20. Demchenko, Y., Laot, C.de., koeroo, Oscar, and Groep, David, 2008. "Re-thinking Grid Security Architecture", in IEEE fourth International conference one Science, Indianapolis, IN USA.

21. Ding, Y., Liu, F. and Tang, B., 2012. Context sensitive trust computing in distributed environments, Knowledge Based Systems, Vol. 28, pp.105-114.

22. Ellison, C.M., Frants, B., Rivest, R., Thomas, B.M. and Ylonen, T., Simple Public Key Certificate, http://www.pobox.com/cme/html/sphci.html.

23. Enokido, T. and Takizawa, M., 2007. A Legal Information Flow (LIF) scheduler for distributed systems, International Conference on Parallel and Distributed Systems, Vol. 2, pp. 1-8.

24. Feng, F., Chuang, L., Peng, D. and Li, J., 2008. A trust and context based access control model for distributed system, 10th IEEE International Conference on High Performance Computing and Communications, pp. 629-634.

25. Firdhous, M., 2011. Implementation of security in distributed system. a comparative study, International Journal of Computer Information Systems, Vol. 2, Issue 2, pp. 1-6.

26. Forouzan, BA, Cryptography and network security, McGraw Hill, 2008.

27. Gasser, M., Goldstein, A., Kaufman, C. and Lampson, B., 1989. The digital distributed system security architecture. Proceedings of the 12th National Computer Security Conference, pages 305-319, 1989.

28. Gehari, A., Labean, T. and Reif, J., 2004. DNA Based Cryptograph, Lecture Notes is Computer Science, Springer.

29. Gollman, D., 1999. Computer Security, John Wiley and Sons. 1999.

30. Gollmann, D., Beth, T. and Damm, F., 1993. Authentication services in distributed systems, Computers and Security, Vol. 12, Issue 8, pp.753-764.

31. Gong, L., 1999 editor. Inside Java 2 platform security architecture, API design, and implementation. Addison – Wesley Longman publishing Co., Inc., 1999.

32. Grandison, T. and Sloman, M., 2000. A survey of trust in Internet applications, IEEE Communications Surveys, Fourth Quarter, (Fourth Quarter), 2000.

33. Hamdi, H. and Mosbah, M., 2009. A DSL framework for policy based security of distributed systems, 3rd IEEE International Conference on Secure Software Integration and Reliability Improvements.

34. Hamdi, H., Bocehula, A. and Mosbah, M., 2007. International Conference on Emerging security Information, systems and technologies.

35. Hau, S. S., Bonatti, P. A., Dengguo, F. and Thuraisingham, B., 2005. Security and privacy in collaborative distributed systems, 29th Annual International Computer Software and Applications Conference.

36. Huang, L. D., Xue, G., He, X. L. and Zhuang, H. L., 2010. A trust model based on evidence theory for P2P systems, Applied Mechanics and Materials, pp. 99-104.

37. Huang, X., Xiang, Y., Chonka, A., Zhou, J. and Deng, R.H, 2011. A generic framework for three factor authentication: Preserving security and privacy in distributed systems, IEEE Transactions on Parallel and Distributed Systems, Vol. 222, Issue 8, pp. 1390-1397.

38. Jansen, W., 1999. Mobile agents and security. NIST. 1999.

39. Jiang, L., Xu, J. and Zhang, K., 2012. A new evidential trust model for open distributed systems, Expert systems with applications, pp. 3772-3782.

40. Josang, A., 2002. Subjective evidential reasoning. In the proceedings of the 9th International Conference on Information Processing and Management of Uncertainty in Knowledge – Based Systems (IPMU 2002), Annecy, France, 1-5 July, 2002.

41. Josang, A., 2001. A logic for uncertain probabilities. International Journal of Uncertainty, Fuzziness and knowledge – based System, 9(3), 279-311, 2001.

42. Kaulman, C., Perlman, R. and Speciner, M., 2002. Network Security: Private communication in a Public world, Prentice Hall Series in computer Networking and distributed system.

43. Kaufmann C, Perlman R, Speciner M, Network security: private communication in a public world, PHI, 2002.

44. Kaur, N., Singh. R., Sarje, A.K. and Misra Manoj, 2005. “Performance evaluation of secure concurrency control algorithm for multilevel secure distributed database system” in International conference on Information Technology coding and computing, Las Vegas, NV USA.

45. Koshutanski, H., 2009. A survey on distributed access control systems for web business process, International Journal of Network Security, Vol. 9, Issue 1, pp. 61-69.

46. Kwoh, Y.K., Song, S. and Hwang, K., 2005. Selfish grid computing: Game-theoretic modeling and nas performance results cardiff. In proceedings of the International Symposium on Cluster Computing and the Grid (CCGrid-2005), Cardiff, UK, May 9-12, 2005.

47. Lange, D.B. and Oshima, M., 1998. Programming and Deploying Ida Mobile Agents with Aglets. Addison – Wesley. 1998.

48. Li, H. and Singhal, M., 2007. Trust Management in distributed systems, Computer, Vol. 40, Issue 2, pp. 45-53.

49. Liao, H., Wang, Q. and Li, G., 2009. A Fuzzy Logic Based Trust Model in Grid, International Conference on Networks Security, Wireless Communications and Trusted Computing NSWCTC, pp. 608-614.

50. Lin, O. and Varadharajan, V., 2003. Modeling and evaluating trust relationship in mobile agent based systems. In proceedings of First International Conference on Applied Cryptography and Network Security (ACNS03), volume LNCS2846. pages 179-190 Kunming, China October 2003, Springer – Verlag, LNCS2846.

51. Lin, C. and Varadharajan, V., 2006. Trust based risk management for distributed system security-a new approach, First International Conference on Availability, Reliability and Security.

52. Lin, C., Varadharajaa, V., Wang, Y. and Pruthi, V., 2005. Trust enhanced security for mobile agents. In 7th International IEEE Conference on E-Commerce Technology 2005, Technische University Mnchen. Germany, July 19-22.2005. IEEE Computer Society Press.

53. Maiden, W., Dionoysiou, I., Frincke, D., Fink, G. and Bakken, D.E., 2011. Dual Trust: A distributed trust model for swarn-based autonomic computing system, DPM 2010 and SETOP 2010, LNCS 6514, pp. 188-202.

54. Marmol, F.G. and Derez, G.M., 2009. Security threats Scenarios in trust and reputation models for distributed systems, Computers and Security, pp. 545-556.

55. Marmol, F.G. and Perez, G. M., 2010. Towards pre-standardization of trust and reputation models for distributed and heterogeneous system, Computer Standards and Interfaces, Vol. 32 (4), pp. 185-196.

56. Marsh, S., 1994. Formalising trust as a computational concept. Ph. D. Thesis. University of Stirling, 1994.

57. Omar, M., Challal, Y. and Bouabdallah, A., 2009. Reliable and fully distributed trust model for mobile ad hoc networks”, Computers and Security, pp. 199-214.

58. Oppliger, R., Grenlich, A. and Trachsel, P., 1999. A distributed certificate management system (DCMS) supporting group based access control, in Proc. 15th IEEE annual computer security application conference (ACSAC’99).

59. Oppliger, R., 1999. Security issues related to mobile code and agent-based systems. Computer Communications, 22(12): 1165-1170, July 1999.

60. Pallickara, S., Ekanayake, J. and Fox, G., 2007. A scalable approach for the secure and authorized tracking of the availability of entities in distributed systems, IEEE International Parallel and distributed Processing symposium, pp. 1-10.

61. Palma, N. De, Hagimont, D., Boyer, F., and Broto, L., 2012. Self protection in a clustered distributed systems, IEEE Transactions on Parallel and Distributed Systems, Vol. 23, Issue 2,pp. 330-336.

62. Piero, R. T. Di, Mancini, L. V. and Mei, A., 2007. Towards threat adaptive dynamic fragment replication in large scale distributed systems, IEEE International Symposium on Parallel and distributed processing, pp. 1-2.

63. Qi, L. and Yu, L., 2001. Mobile agent based security model for distributed system, IEEE International Conference on Systems, Man and Cybernetics.

64. Rasmusson, L. and Jansson, S., 1996. Simulated social control for secure internet commerce: Position paper at the new security paradigms workshop 1996.

65. Schryen, G., Volkemer, M., Ries, S. and Habib, S.M., 2011. A formal approach towards measuring trust in distributed systems, ACM Symposium on Applied Computing (SAC'11), pp.1739-1745.

66. Seamons, K. and Winsbotough, W., 2002. Automated trust Negotiation Technical Report, Us Patent and Trade Mark office, IBM Corporation, Patent application field Max7.

67. Seung, W.J. and Souhan, J., 2006. Secure Password authentication for distributed computing, International Conference on Computational Intelligence and Security, Vol.2, pp.1345-1350.

68. Shehab, M., Ghafoor, A. and Bertino, E., 2010. Secure collaboration in a mediator free distributed environment, IEEE Transactions on Parallel and Distributed Systems, Vol. 19, Issue 10, pp. 1338-1351.

69. Shenbagavadivu, N. and Usha Savithri, S., 2012. Enhanced Information security in distributed mobile system based on delegate object model, Procedeia Engineering, Vol. 30, pp. 774-781.

70. Song, S. and Hwang, K., 2004. Fuzzy trust integration for security enforcement in grid computing. In International Symposium on Network and Parallel Computing (NPC2004), submitted March 22, 2004.

71. Sonntag, M. and Hrmanseder, R., 2000. Mobile agent security based on payment. Operating Systems Review, 34(4): 48-55, 2000.

72. Stallings, W, Network and internetwork security principles and practice, Prantic Hall, 2004.

73. Tang, W. and Chen, Z., 2003. Research of subjective trust management model based on the fuzzy set theory, Journal of Software, Vol. 14(8), pp. 1401-1408.

74. Tomoya, E. and Makoto, T., 2011. Con-currency control based on significance on roles; 11th International Conference on Parallel and Distributed Systems.

75. Ukil and Arijit, 2011. Secure Trust Management in Distributed Computing Systems, Sixth IEEE International Symposium on Electronic Design, Test and Application (DELTA), pp. 116-121.

76. Uzunov, A.V., Fernandez, E.B. and Falkner, K., 2012. Securing distributed systems using patterns: a survey, Computers and Security, Vol. 31(5), pp. 681-703.

77. Varadhanrajan, V., 2000. Security enhanced mobile agents. Proc. of 7th ACM Conference on Computer and Communication Security, 2000.

78. Vhoi, J. Y., Li, Z. Y., Yaun, H. Y. and Song, O., 2011. Privacy protection in service discovery for large scale distributed computing systems, IEEE International Symposium on Parallel and Distributed Processing Workshops and Ph. D. Forum (IPDPSW), pp.1025-1032.

79. Vieira, K., Schulter, A., Westphall, C. B. and Westphall, C. M., 2010. IT professional, Vol. 12 Issue 4, pp. 38-43.

80. Wang, J. and Sun, H. J., 2009. A new evidential trust model for open communities, Computer Standards and Open Interfaces, pp.994-1001.

81. Wietrzyk, V. I., Tajuzawa, M., Orgun, M. A. and Varadharajann, V., 2001. A secure transaction environment for work flows in distributed systems, 8th International Conference on Parallel and Distributed Systems.

82. Xiaoyong, T., Li, K., Zong, Z. and Veeravalli, B., 2011. A novel security-driven scheduling algorithms for precedence-constrained tasks in heteroge-neous distributed systems, IEEE Transactions on Computers, Vol. 60, Issue 7, pp. 1017-1029.

83. Xie, T. and Qin, X., 2007. Performance evaluation of a new scheduling algorithm for distributed systems with security heterogeneity, Journal of Parallel and Distributed Computing, Vol. 67, Issue 10, pp.1067-1081.

84. Xie, T. and Qin, X., 2008. Securing aware resource allocation for real time parallel jobs on homogeneous and heterogeneous clusters, IEEE Transactions on parallel and Distributed System, Vol. 19, Issue 5, pp. 682-697.

85. Xu, Y., Korba, L., Wang, L., Hao, Q., Shen, W. and Lang, S., 2003. A security framework for collaborative distributed system control at the device level, IEEE International Conference on Industrial Informatics.

86. Yao, W. and Fidelis, 2003. A policy driven trust management framework in iTrust, LNCS 2692, pp. 301-314, Springer-Verlag.

87. Yu, B. and Singh, M. P., 2002. An evidential model of distributed reputation management, First International Joint Conference on Automous Agents and Multiagent Systems.

88. Yu, B. and Singh, M.P., 2002. Distributed reputation management for electronic commerce. First International Joint Conference on Autonomous Agents Multiagent Systems, Bologna. Italy, 2002.

89. Zadeh, L A, 1965, Fuzzy Sets, Information and Control, Vol 65, pp. 27-76.

90. Zhao, Y. and Thomas, N., 2009. Computing methods for efficient analysis of PEPA models of non-repudiation protocols, 15th International Conference on Parallel and Distributed Systems.

91. Zhou, H., Meng, X., Zhang, L., and Oiao, X., 2010. Quorum systems for intrusion tolerance based on trusted timely computing base, Journal of Systems, Engineering and Electronics, Vol. 21, Issue 1, pp.168-174.

92. Zhou, X., 2012. A modelling approach using UML2 for security protocols in distributed systems, LNCS, Vol. 141, pp. 57-64.

93. Zong, B., Xu, F., Pan, J. and Lv. J., 2009. Comparing and evaluating collection strategizes in trust based reputation system in distributed environment, symposia and workshops on ubiquitous, Autonomic and Trusted Computing, pp. 557-562.

94. Zubi, Z. S., 2009. On distributed database security aspects in International conference on Multimedia computing and System, Ouarzazate, Moracco.