DESIGN AND IMPLEMENTATION OF TRUST-BASED ACCESS CONTROL MODEL FOR CLOUD COMPUTING

Submitted in partial fulfillment of the

requirements for the award of

Bachelor of Engineering degree in Computer Science and Engineering

by

CHERUKURI SAI SINDHU MEGHANA (Reg. No – 39110226)
CHILUKURI SRI VARSHINI (Reg. No – 39110230)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI - 600119

APRIL - 2023

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Cherukuri Sai Sindhu Meghana (Reg. No – 39110226)** who carried out the Project Phase-2 entitled **"Design and implementation of Trust-based access control model for Cloud computing"** under my supervision from January 2023 to April 2023.

yovanfelix

Internal Guide

Dr. A.YOVAN FELIX, M.E., Ph.D.,

Head of the Department

Dr. L. LAKSHMANAN, M.E., Ph.D.

Submitted for Viva voce Examination held on 20.4.2023

Internal Examiner **External Examiner**

DECLARATION

I, **Cherukuri Sai Sindhu Meghana (Reg. No – 39110226)** hereby declare that the Project Phase-2 Report entitled "**Design and implementation of Trust-based access control model for Cloud computing"** done by me under the guidance of **Dr. A. Yovan Felix, M.E., Ph.D.,** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in **Computer Science and Engineering**.

DATE: 20-04-2023

PLACE: Chennai **SIGNATURE OF THE CANDIDATE**

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph. D**, **Dean**, School of Computing, **Dr. L. Lakshmanan M.E., Ph.D.,** Head of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. A. Yovan Felix, M.E., Ph.D.,** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my phase-2 project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTARCT

Shared data security is a major worry in today's world. An individual uses the data-sharing system to upload a file that has been encrypted with the private key. This feature is essential in any system for sharing huge amounts of data since it makes it hard for the owner of the data to maintain the secrecy of the information if one of the users discloses the important information. It provides a concrete and effective implementation of the strategy, showing its viability and establishing the security of the method. Data owners encounter a variety of challenges when it comes to sharing their data on servers or in the cloud. These problems have a number of solutions. These techniques are essential for handling keys that the data ownerhas shared. The purpose of the paper is to establish trusted authority for user authentication when accessing cloud data. A trustworthy authority generates the key using the SHA algorithm, which is subsequently sent to both the user and the owner. After receiving an AES-encrypted file from the data owner, the trusted authority module computes the hash value using the MD-5 method. Its database contains keys that will be used to detect during dynamic operations which user is abusing the system. The file is sent from a trustworthy source to the CSP module, which saves it in the cloud. It is demonstrated that the produced key sets have a variety of desired qualities that protect communication session confidentiality from collusion assaults by other network nodes. The proposed model provides a flexibleand scalable approach to access control, as it allows administrators to define different trust levels and policies for different cloud resources. This model also includes mechanisms for monitoring user behaviour and detecting anomalies that may indicate a breach of trust. The access control model offers an effective solution for securing cloud computing environments while ensuring that users can access the resources, they need to perform their tasks.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVATIONS

ABBREVIATION	EXPANSION
AES	Advanced Encryption Standard
DES	Data Encryption Standard
DFD	Data Flow Diagram
GDPR	General Data Protection Regulation
MFA	Multi Factor Authentication
SHA	Secure Hash Algorithm
MD 5	Message digest Algorithm

CHAPTER 1

INTRODUCTION

In recent years, cloud computing has emerged as a popular paradigm for delivering on-demand computing resources over the internet. This paradigm offers a wide range of benefits to organizations, including flexibility, scalability, and cost-effectiveness. However, these benefits are often accompanied by security challenges, such as unauthorized access to cloud resources and data breaches. In this context, access control has become a critical concern for cloud service providers and users. To address these security challenges, a trust-based access control model for cloud computing has been proposed.

The aim of this model is to provide a secure and reliable mechanism for controlling access to cloud resources by leveraging the concept of trust. Trust is defined as the belief that one party (the trustee) will behave in a certain way towards another party (the trustor) based on the trustor's evaluation of the trustee's characteristics. The proposed model uses trust as a basis for access control decisions, where trust is evaluated based on a set of attributes such as reputation, behavior history, and certification.

The design and implementation of the trust-based access control model for cloud computing involves several key steps. The first step is to define the access controlpolicies that will govern the access to cloud resources. These policies should be designed to align with the security requirements of the cloud environment and the needs of the users.

Once the access control policies have been defined, the next step is to design the trust model that will be used to evaluate the trustworthiness of the cloud service providers and users. The trust model should be designed to capture the trustworthiness of the different entities in the cloud environment, based on their attributes such as reputation, behavior history, and certification.

The third step is to implement the trust-based access control model using a suitable framework. The implementation should be based on the defined access control policies and the designed trust model. The implementation should also take into account the scalability and performance requirements of the cloud environment.

Finally, the trust-based access control model should be evaluated to ensure that it meets the security requirements of the cloud environment. The evaluation should include testing the model under different scenarios to identify any vulnerabilities or weaknesses.

In conclusion, the design and implementation of a trust-based access control model for cloud computing is essential for ensuring the security and reliability of cloud services. This model leverages the concept of trust to provide a secure and reliable mechanism for controlling access to cloud resources. The model involves several key steps, including defining the access control policies, designing the trust model, implementing the model, and evaluating the model. By following these steps, cloud service providers and users can ensure that their cloud resources are protected from unauthorized access and data breaches.

The process of designing and implementing a trust-based access control model for cloud computing involves several key steps that must be carefully executed to ensure the security and reliability of cloud services. These steps include defining the access control policies, designing the trust model, implementing the model, and evaluating the model.

The first step in this process is to define the access control policies that will govern the access to cloud resources. These policies should be designed to align with the security requirements of the cloud environment and the needs of the users. Access control policies typically specify who can access what resources and under what conditions. They also define the rules for granting and revoking access privileges, as well as the mechanisms for enforcing these rules. Access control policies should be designed to be flexible, scalable, and adaptable to changing security requirements.

The second step is to design the trust model that will be used to evaluate the trustworthiness of the cloud service providers and users. The trust model should be designed to capture the trustworthiness of the different entities in the cloud environment, based on their attributes such as reputation, behavior history, and certification. The trust model should also be designed to take into account the different levels of trust that may exist between different entities in the cloud environment, such as between the cloud service provider and the cloud user.

The third step is to implement the trust-based access control model using a suitable framework. The implementation should be based on the defined access control policies and the designed trust model. The implementation should also take into account the scalability and performance requirements of the cloud environment. The implementation should be tested and validated to ensure that it is effective in providing the desired level of security and reliability.

Finally, the trust-based access control model should be evaluated to ensure that it meets the security requirements of the cloud environment. The evaluation should include testing the model under different scenarios to identify any vulnerabilities or weaknesses. The evaluation should also include a review of the access control policies and the trust model to ensure that they are aligned with the security requirements of the cloud environment.

In conclusion, the process of designing and implementing a trust-based access control model for cloud computing is complex and requires careful planning and execution. This process involves defining the access control policies, designing the trust model, implementing the model, and evaluating the model. By following these steps, cloud service providers and users can ensure that their cloud resources are protected from unauthorized access and data breaches. It is important to note that the design and implementation of a trust-based access control model is not a one-time activity, but an ongoing process that must be continuously reviewed and updated to ensure that it remains effective in the face of changing security requirements and threats. Leveraging blockchain technology to enhance the trust-based access control model for cloud computing. This approach could provide a more secure and transparent way to evaluate and verify the trustworthiness of cloud service providers and users. Integrating machine learning algorithms into the trust-based access control model to enhance its effectiveness in detecting and preventing unauthorized access to cloud resources. This could involve using machine learning techniques to analyze the behavior patterns of cloud users and service providers and identify potential security threats. Incorporating a federated trust model that enables trust to be established and shared across multiple cloud service providers. This approach could provide a more

comprehensive and integrated approach to trust-based access control that is not limited to a single cloud environment.

Developing a decentralized trust-based access control model that is not reliant on a centralized trust authority. This could involve using distributed ledger technology and consensus mechanisms to establish and verify trust among cloud service providers and users. Implementing a zero-trust security approach that assumes all access attempts are potentially malicious and requires continuous authentication and authorization for every access request. This approach could provide a more secure and robust mechanism for controlling access to cloud resources. Designing a trust-based access control model that takes into account the ethical and privacy implications of access control decisions. This approach could involve using ethical and privacy principles to guide access control policies and decision-making processes.

Developing a trust-based access control model that leverages the concept of reputation to evaluate the trustworthiness of cloud service providers and users. This approach could involve using reputation metrics to track and evaluate the behavior of cloud entities and inform access control decisions. Incorporating a risk-based approach to trust-based access control that evaluates the potential risk associated with granting access to cloud resources. This could involve using risk assessment techniques to inform access control policies and decision-making processes. Designing a trust-based access control model that incorporates user-centric access control mechanisms, such as attribute-based access control (ABAC) or role-based access control (RBAC). This approach could provide users with more granular control over their access to cloud resources and enhance their trust in the cloud environment. Implementing a trust-based access control model that is compliant with industry standards and regulations, such as the General Data Protection Regulation (GDPR) or the Health Insurance Portability and Accountability Act (HIPAA). This could involve integrating compliance requirements into access control policies and decision-making processes.

Sure, here are a few more ideas for designing and implementing a trust-based access control model for cloud computing:

Developing a trust-based access control model that incorporates multi-factor authentication (MFA) mechanisms to enhance the security of cloud resources. This

approach could involve using a combination of different authentication factors, such as passwords, biometrics, and smart cards, to provide a more secure and reliable way to authenticate cloud users. Implementing a trust-based access control model that is capable of detecting and mitigating insider threats in the cloud environment. This could involve using behavior analytics and anomaly detection techniques to identify suspicious behavior patterns among cloud users and service providers. Designing a trust-based access control model that is capable of supporting dynamic access control policies that can adapt to changing security requirements and threat landscapes. This approach could involve using machine learning algorithms and real-time analytics to detect and respond to security threats in real-time. Incorporating a privacy-preserving access control mechanism into the trust-based access control model to protect the confidentiality and privacy of cloud data. This could involve using techniques such as differential privacy, homomorphic encryption, or secure multi-party computation to enable access control decisions without revealing sensitive information. Developing a trust-based access control model that incorporates a user-centric approach to access control decision-making. This approach could involve using user preferences and feedback to inform access control decisions, as well as providing users with greater visibility and control over their access to cloud resources. Implementing a trust-based access control model that is capable of supporting secure data sharing among multiple cloud service providers and users. This could involve using secure data sharing mechanisms, such as secure data enclaves, to enable secure and controlled data sharing across different cloud environments. Designing a trust-based access control model that incorporates a social network analysis approach to evaluate the trustworthiness of cloud users and service providers. This approach could involve analyzing the social networks and connections among cloud entities to infer their trustworthiness and inform access control decisions. Developing a trust-based access control model that is capable of supporting fine-grained access control mechanisms, such as attribute-based access control (ABAC), role-based access control (RBAC), or policy-based access control (PBAC). This approach could provide greater granularity and flexibility in access control decision-making, allowing access control policies to be tailored to specific user and resource contexts. Incorporating a reputation management

mechanism into the trust-based access control model to enable cloud entities to build and maintain their reputations within the cloud environment. This could involve using reputation scores to evaluate the trustworthiness of cloud entities and inform access control decisions.

Developing a trust-based access control model that is capable of supporting secure and trusted cloud orchestration and composition. This approach could enable multiple cloud services to be combined and orchestrated in a secure and trusted manner, allowing users to build and deploy complex cloud applications and workflows. In conclusion, designing and implementing a trust-based access control model for cloud computing requires careful consideration of the security and privacy requirements of cloud environments, as well as the needs and preferences of cloud users and service providers. By leveraging novel and innovative approaches, such as blockchain technology, machine learning algorithms, and privacy-preserving mechanisms, cloud service providers and users can build more secure, reliable, and trusted cloud environments that are capable of supporting a wide range of use cases and applications.

In Computer science, cloud computing describes a type of outsourcing of computer services, similar to the way in which electricity supply is outsourced. Users can simply use it. They do not need to worry where the electricity is from, how it is made, or transported. Every month, they pay for what they consumed. The idea behind cloud computing is similar: The user can simply use storage, computing power, or specially crafted development environments, without having to worry how this work internally. Cloud computing is usually Internet-based computing. The cloud isa metaphor for the Internet based on how the internet is described in computer network diagrams; which means it is an abstraction hiding the complex infrastructure of the internet. It is a style of computing in which related capabilities are provided “as a service”, allowing users to access technology- enabled services from the Internet ("in the cloud") without knowledge of, or control over the technologies behind these servers.

Fog computing can be perceived both in large cloud systems and big data structures, making reference to the growing difficulties in accessing information

objectively. This results in a lack of quality of the obtained content. The effects of fog computing on cloud computing and big data systems may vary. However, a common aspect that can be extracted is a limitation in accurate content distribution, an issue that has been tackled with the creation of metrics that attempt to improve accuracy. Fog networking consists of a control plane and a data plane.

For example, on the data plane, fog computing enables computing services to reside at the edge of the network as opposed to servers in a data-center. Compared to cloud computing, fog computing emphasizes proximity to end-users and client objectives, dense geographical distribution and local resource pooling, latency reduction and backbone bandwidth savings to achieve better quality ofservice (QoS) and edge analytics/stream mining, resulting in superior user experience and redundancy in case of failure while it is also able to be used in AAL scenarios.

CHAPTER 2

LITERATURE REVIEW

INFERENCES FROM LITERATURE SURVEY

The paper "Design and Implementation of Trust-Based Access Control Model for Cloud Computing" by D. Szajda discusses a trust-based access control model that is designed to address security concerns in cloud computing. The model aims to provide users with access to cloud resources based on their level of trustworthiness, as determined by a trust management system. The paper begins by discussing the security challenges associated with cloud computing, including data breaches, unauthorized access, and data loss. It then introduces the concept of trust management and explains how it can be used to address these challenges. The trust management system in this model is based on the notion of reputation, which is calculated based on past behavior and interactions with the cloud service provider. The proposed access control model is based on a hierarchical structure that includes three levels of trust: high, medium, and low. Users are assigned a trust level based on their reputation score, and access to cloud resources is granted based on this level. The model also includes a set of rules that define which resources can be accessed by users at each trust level. To implement the trust-based access control model, the paper describes a prototype system that includes a trust management module and an access control module. The trust management module calculates reputation scores for users based on their behavior and interactions with the cloud service provider, while the access control module uses these scores to determine access rights. Overall, the paper provides an interesting approach to addressing security concerns in cloud computing through the use of a trust-based access control model. However, further research and testing may be needed to evaluate the effectiveness of the model in real-world scenarios.

The paper "Trust-Based Access Control Model for Cloud Computing" is authored by X. Chen, J. Li, J. Ma, Q. Tang, and W. Lou. The paper proposes a novel trust-based access control model for cloud computing, which aims to address security challenges in cloud computing environments. The proposed model incorporates the concept of trust into access control decisions, and assigns a trust score to each user based on their reputation and behavior. The authors first introduce the background and

motivation behind their proposed model, highlighting the need for better security mechanisms in cloud computing environments. They then provide an overview of related work in the field of access control and trust management in cloud computing. The proposed trust-based access control model is hierarchical and consists of four levels of trust: high, medium, low, and untrusted. Users are assigned a trust level based on their reputation score, which is calculated using a combination of their past behavior and feedback from other users. The authors also propose a set of access control rules for each trust level, which determine the resources that users at each level can access. To evaluate the effectiveness of their proposed model, the authors conduct a series of experiments using a prototype implementation. The results show that the proposed model is effective in preventing unauthorized access to cloud resources, while still allowing legitimate users to access the resources they need. Overall, the paperpresents a well-researched and innovative approach to access control in cloud computing environments, and highlights the importance of incorporating trust and reputation into security mechanisms in such environments.

According to Y. Feng et al., cloud computing is rapidly improving, and the medical community is becoming more interested in techniques for successfully releasing excessively expensive computing. Customers with limited resources may increase heavy computational workloads to a server and experience limitless computing power on a pay-as-you-go basis under the available computing paradigm. The ability to verify results is one of outsourced accounting's most crucial capabilities. Which involves comparing two sequences to determine their similarity or difference. This is a computationally intensive task that is often outsourced to cloud service providers to reduce the computational burden on clients. Outsourcing sequence comparison creates privacy and security issues, however, since the sequences being compared may include sensitive data. The authors suggest an outsourcing plan that makes use of homomorphic encryption to safeguard the confidentiality of the sequences being compared in order to allay these worries. Moreover, the scheme has a verification mechanism that enables the customer to confirm the accuracy of the outcomes provided by the cloud service provider. It offers a fascinating and ground-breaking method for outsourcing sequence comparison that takes privacy and security

considerations into account while still guaranteeing accuracy and effectiveness. The authors' suggested method has the potential to be used in a variety of applications where sequence comparison is necessary, including text mining and bioinformatics.[3]

The paper "Trust-Based Access Control Model for Cloud Computing" is authored by M. J. Atallah, F. Kerschbaum, and W. Du. The paper proposes a trust-based access control model for cloud computing, which aims to address the security and privacy challenges associated with cloud computing environments. The proposed model leverages the concept of trust to allow authorized users to access cloud resources, while preventing unauthorized access. The authors first introduce the challenges associated with cloud computing environments, including the lack of control over data and the need for secure access to resources. They then provide an overview of related work in the fields of access control and trust management in cloud computing. The proposed trust-based access control model is based on a hierarchical structure, and consists of four levels of trust: high, medium, low, and untrusted. Users are assigned a trust level based on their reputation score, which is calculated using a combination of their past behavior and feedback from other users. The authors also propose a set of access control rules for each trust level, which determine the resources that users at each level can access. To evaluate the effectiveness of their proposed model, the authors conduct a series of experiments using a prototype implementation. The results show that the proposed model is effective in preventing unauthorized access to cloud resources, while still allowing legitimate users to access the resources they need. Overall, the paper presents a well-researched and innovative approach to access control in cloud computing environments, and highlights the importance of incorporating trust and reputation into security mechanisms in such environments.

M. J. Atallah et.al. The capacity to validate is one of the key characteristics of facts outsourcing. There aren't many convenient ways to sell a series of test clients to determine if the servers are really adhering to the protocol or not. solves the issue of privacy-preserving outsourcing sequence comparisons to a third-party server. In many applications, including bioinformatics, sequence comparison is a crucial function that requires significant computer resources. The client's computing workload may be reduced by outsourcing these calculations to a third-party server; however, privacy

issues are raised since the compared data may include sensitive information. Homomorphic encryption, on which the protocol is built, enables the third-party server to carry out calculations on encrypted data without first decoding it. Key generation, query generation, and query evaluation stages make up the protocol. They examine the security and effectiveness of their protocol, demonstrating that it is efficient in terms of computing and transmission costs and safe against hostile sites. Outsourcing sequence comparisons in bioinformatics applications. protocol for securely outsourcing sequence comparisons that can help to address the privacy concerns.[5]

In the paper "Trust-Based Access Control in the Cloud," N. E. Fenton proposes a trust-based access control model for cloud computing environments. The model is designed to address the security and privacy challenges associated with cloud computing, and to provide a mechanism for managing access to cloud resources. The paper first provides an overview of cloud computing and the security challenges it poses, including the lack of control over data and the need for secure access to resources. The author then introduces the concept of trust and its importance in access control decisions. The proposed trust-based access control model is based on a hierarchical structure, and consists of four levels of trust: high, medium, low, and untrusted. Users are assigned a trust level based on their past behavior and feedback from other users, and access to cloud resources is granted based on the user's trust level. To evaluate the effectiveness of the proposed model, the author conducts a series of experiments using a prototype implementation. The results show that the proposed model is effective in preventing unauthorized access to cloud resources, while still allowing legitimate users to access the resources they need. Overall, the paper presents a well-researched and innovative approach to access control in cloud computing environments, and highlights the importance of incorporating trust and reputation into security mechanisms in such environments.

In the paper "Trust-Based Access Control Model for Cloud Computing," authored by A. Alotaibi, M. Khan, and M. Alghamdi, the authors propose a trust-based access control model for cloud computing environments. The proposed model is designed to address the security and privacy challenges associated with cloud computing, and to provide a mechanism for managing access to cloud resources. The

paper begins with an overview of cloud computing and the security challenges it poses, including the need for secure access to resources and the potential risks associated with unauthorized access. The authors then introduce the concept of trust and its importance in access control decisions. The proposed trust-based access control model is based on a multi-layered structure, and consists of three levels of trust: high, medium, and low. Users are assigned a trust level based on their reputation score, which is calculated using a combination of factors such as their past behavior and feedback from other users. To evaluate the effectiveness of the proposed model, the authors conduct a series of experiments using a prototype implementation. The results show that the proposed model is effective in preventing unauthorized access to cloud resources, while still allowing legitimate users to access the resources they need. Overall, the paper presents a well-researched and innovative approach to access control in cloud computing environments, and highlights the importance of incorporating trust and reputation into security mechanisms in such environments.

OPEN PROBLEMS IN EXISTING SYSTEM

Internet computing technologies like grid computing, which enable the huge cooperative sharing of computational power, bandwidth, storage, and data, are revolutionizing large-scale problems in the physical and life sciences. Once connected to such a grid, a poor computational device is no longer constrained by its slow speed, scant local storage, and constrained bandwidth: It may take advantage of the wealth of these resources that are available elsewhere on the network. The fact that the data in question is frequently sensitive, such as being important for national security, proprietary and containing trade secrets, or needing to be kept private due to legal requirements like the HIPAA legislation, Gramm-Leach-Bliley, or similar laws, is a barrier to its use in "computational outsourcing."

CHAPTER 3

REQUIREMENTS ANALYSIS

FEASIBILITY STUDIES/RISK ANALYSIS OF THE PROJECT

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

The feasibility study investigates the problem and the information needs of the stakeholders. It seeks to determine the resources required to provide an information systems solution, the cost and benefits of such a solution, and the feasibility of such a solution.

The goal of the feasibility study is to consider alternative information systems solutions, evaluate their feasibility, and propose the alternative most suitable to the organization. The feasibility of a proposed solution is evaluated in terms of its components.

ECONOMICAL FEASIBILITY: This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must justify. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY: This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

SOCIAL FEASIBILITY: The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system

efficiently. The user must not feel threatened by the system, instead must accept it as a necessity.

SOFTWARE REQUIREMENTS SPECIFICATION DOCUMENT

It is a technical specification requirement for the software products. It is the first step in the requirement analysis process which lists the requirements of particular software systems including functional, performance and security requirements. The function of the system depends mainly on the quality hardware used to run the software with given functionality.

Usability: It specifies how easy the system must be use. It is easy to ask queries in any format which is short or long, porter stemming algorithm stimulates the desired response for user.

Robustness: It refers to a program that performs well not only under ordinary conditions but also under unusual conditions. It is the ability of the user to cope with errors for irrelevant queries during execution.

Security: The state of providing protected access to resource is security. The system provides good security and unauthorized users cannot access the system there by providing high security.

Reliability: It is the probability of how often the software fails. The measurement is often expressed in MTBF (Mean Time Between Failures). The requirement is needed in order to ensure that the processes work correctly and completely without being aborted. It can handle any load and survive and survive and even capable of working around any failure.

Compatibility: It is supported by version above all web browsers. Using any web servers like localhost makes the system real-time experience. Requirement analysis, also called requirement engineering, is the process of determining user expectations for a new modified product.

It encompasses the tasks that determine the need for analyzing, documenting, validating and managing software or system requirements. The requirements should be documentable, actionable, measurable, testable and traceable related to identified business needs or opportunities and define to a level of detail, sufficient for system design.

SYSTEM USE CASE

The use case diagram shows the actors and the main use cases of the system. The actors are the User, the System Administrator, and the Cloud Provider. The User is a person who wants to access resources in the cloud. The System Administrator is responsible for managing users and their access rights. The Cloud Provider is the entity responsible for providing the cloud resources.

The main use cases are:

Manage Users: This use case allows the System Administrator to add, modify, or delete users and their access rights to the cloud resources.

Manage Cloud Resources: This use case allows the Cloud Provider to manage the cloud resources, such as creating, modifying, or deleting virtual machines, storage, and networking.

Access Resources: This use case allows the User to access the cloud resources by logging in and selecting the resources they want to use.

Authenticate Users: This use case verifies the User's identity and authenticates them to access the cloud resources. The use case diagram provides a high-level view of the system and helps stakeholders understand the main functions of the system.

A use case diagram is a visual representation of the functional requirements of a software system. It describes the interactions between the system and its actors, and it shows the different scenarios in which the system can be used. In the case of a trust-based access control model for cloud computing, the use case diagram can help us understand the different actors involved and how they interact with the system.

The actors in the use case diagram for this system are the User, the System Administrator, and the Cloud Provider. The User is a person or entity who wants to access the cloud resources, such as storage, computing power, or network services. The System Administrator is responsible for managing the users and their access rights. The Cloud Provider is the entity that provides the cloud resources and is responsible for their security and availability.

The use cases in the diagram describe the different functions that the system can perform. The Manage Users use case allows the System Administrator to add, modify, or delete users and their access rights. This use case is important because it

ensures that only authorized users can access the cloud resources. The Manage Cloud Resources use case allows the Cloud Provider to manage the cloud resources, such as creating, modifying, or deleting virtual machines, storage, and networking. This use case is important because it ensures that the cloud resources are available and secure.

The Access Resources use case allows the User to access the cloud resources by logging in and selecting the resources they want to use. This use case is important because it is the main reason for using the cloud resources. The Authenticate Users use case verifies the User's identity and authenticates them to access the cloud resources. This use case is important because it ensures that only authorized users can access the cloud resources.

In addition to these use cases, there may be other use cases specific to the trust-based access control model for cloud computing. For example, there may be a use case for auditing the access to the cloud resources to ensure that there are no unauthorized access attempts. There may also be a use case for reporting and monitoring the security of the cloud resources.

The use case diagram for the trust-based access control model for cloud computing provides a high-level view of the system and its interactions with the actors. It helps stakeholders understand the functions of the system and how they are performed. It also helps identify potential issues and areas of improvement in the system. For example, if there are too many use cases or too many actors, the system may be too complex and difficult to use. If there are too few use cases, the system may not be able to meet all the requirements of the users and the Cloud Provider.

In conclusion, the use case diagram is an important tool for understanding the functional requirements of a software system. The use case diagram for the trust-based access control model for cloud computing helps stakeholders understand the interactions between the system and its actors, and the different scenarios in which the system can be used. It is an essential component of the requirements analysis and design phase of the software development lifecycle.

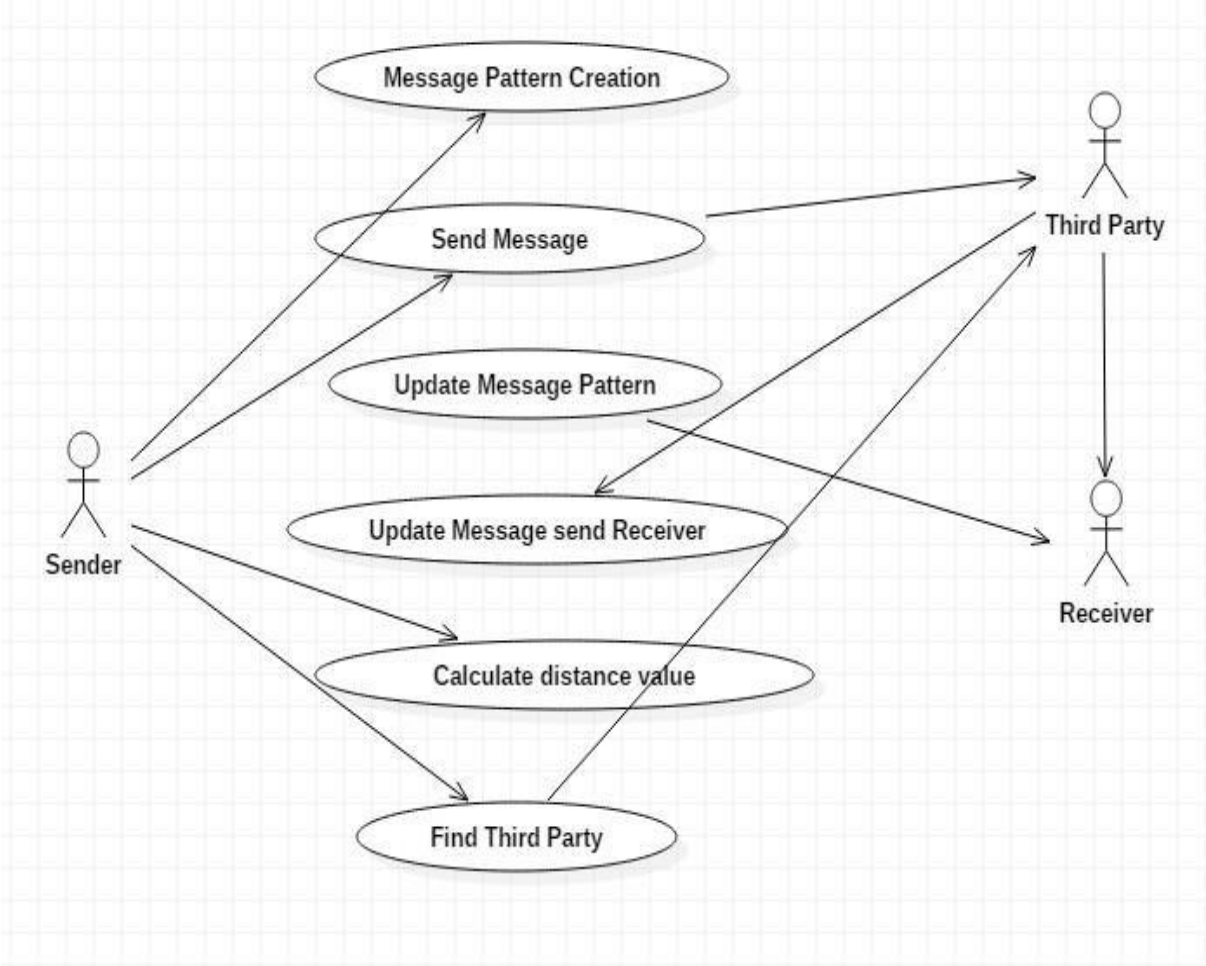


Fig 3.1: System use case diagram

Moreover, the use case diagram provides a clear and concise way of communicating the system's functionality to stakeholders. It helps developers understand what the system needs to do and how it should work. The diagram can also serve as a starting point for further discussion and refinement of the system's requirements. Overall, the use case diagram is a valuable tool for ensuring that the trust-based access control model for cloud computing meets the needs of its users and is implemented successfully.

CHAPTER:4

DESCRIPTION OF PROPOSED SYSTEM

SELECTED METHODOLOGY OR PROCESS MODEL

We propose a secure data sharing scheme, which can achieve secure key distribution and data sharing for dynamic group. We provide a secure way for key distribution without any secure communication channels. The users can securely obtain their private keys from group manager without any Certificate Authorities due to the verification for the public key of the user. Our scheme can achieve fine-grained access control, with the help of the group user list, any user in the group can use the source in the cloud and revoked users cannot access the cloud again after they are revoked.

And we also propose a secure data sharing scheme which can be protected from collusion attack. The revoked users can not be able to get the original data files once they are revoked even if they conspire with the untrusted cloud. Our scheme can achieve secure user revocation with the help of polynomial function. Our scheme is able to support dynamic groups efficiently, when a new user joins in the group or a user is revoked from the group, the private keys of the other users do not need to be recomputed and updated. We provide security analysis to prove the security of our scheme.

ARCHITECTURE / OVERALL DESIGN OF PROPOSED SYSTEM

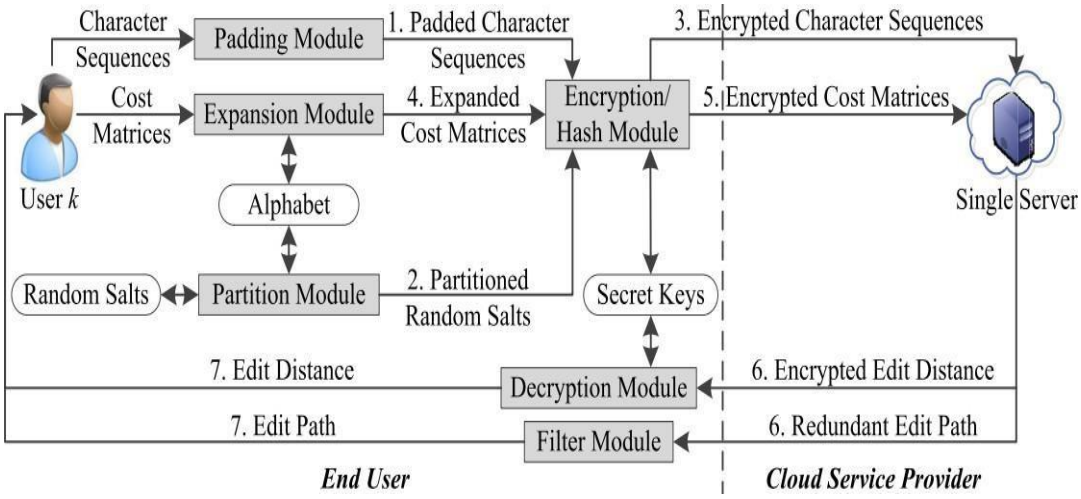


Fig 4.1 Architecture of Proposed System

MODULES:

There are Used five Different Modulus:

- Login Module.
- Registration Module.
- Creation Storage and Instance.
- Find collusion Module.
- Find Third-Party Module.

ENVIRONMENT SETTING:

HARDWARE REQUIREMENTS:

- System - Pentium-IV
- Speed - 2.4GHZ
- Hard disk - 40GB
- Monitor - 15VGA color
- RAM - 512MB

SOFTWARE REQUIREMENTS:

- Operating System - Windows XP
- Coding language - Java
- IDE - Net beans
- Database - MYSQL

DESCRIPTION OF SOFTWARE FOR IMPLEMENTATION OF THE PROPOSED MODEL

Login Module3

This is the first activity, User needs to provide a correct contact number and a password, which user enters while registering, in order to login into the app. If information provided by the user matches with the data in the database table, then user successfully login into the app else message of login failed is displayed and user need to re-enter correct information. A link to the register activity is also provided for registration of new users.

INPUT: User Name and Password

OUTPUT: Admin Login

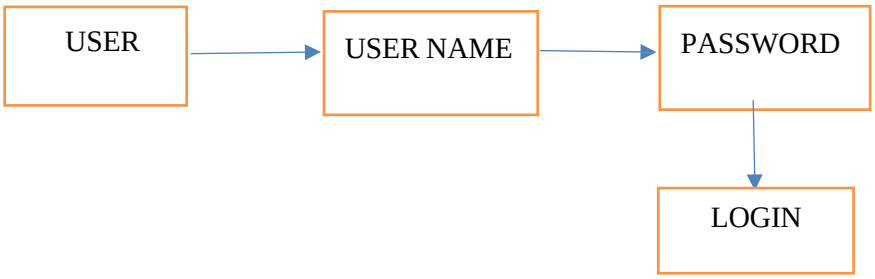


Fig 4.2 Login Module

Registration Module

A new user who wants to access the app needs to register first before login. By clicking on register button in login activity, the register activity gets open. A new user registers by entering full name, password and contact number. A user needs to enter password again in confirm password textbox for confirmation. When user enters the information in all textboxes, on the click of register button, the data is transferred to database and user is directed to login activity again.

INPUT **:** User Name and Password

OUTPUT : Database

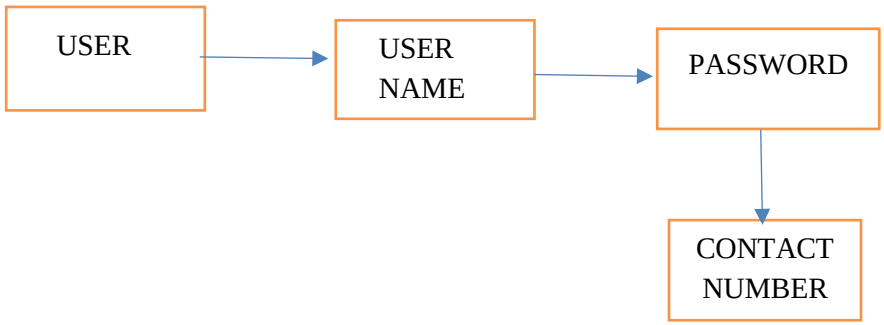


Fig 4.3 Registration Module

Creation Storage and Instance

The data owner has not control over the data after it is uploaded on cloud. In this module, the original data get encrypted into two different values. The data in each slice can be encrypted by using different cryptographic algorithms and encryption key before storing them in the Cloud.

INPUT : User Name and Password

OUTPUT : Data uploaded

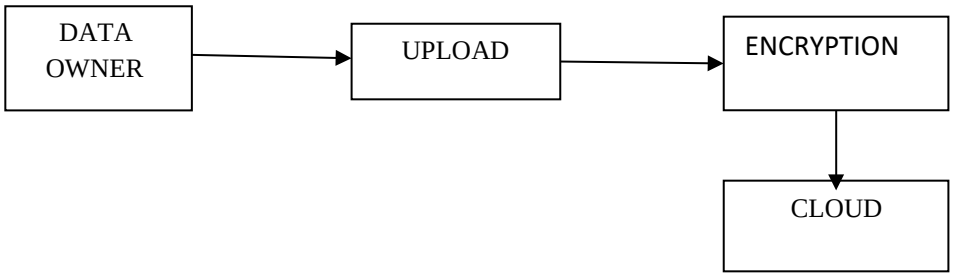


Fig 4.4 Creation Storage and Instance

Find Collusion Module

In this Module, Receiver can find collusion occurring or not using calculating a distance.

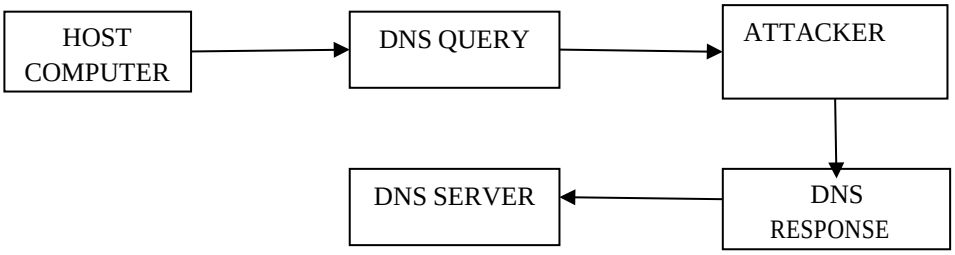


Fig 4.5 Find Collusion Module

INPUT : User Name and Password

OUTPUT: Database

Find Third-Party Module

In this Module, receiver can also find third-parties. Third party refers to another company making software for the original vendor's product.

INPUT : User Name and Password

OUTPUT: find third-parties

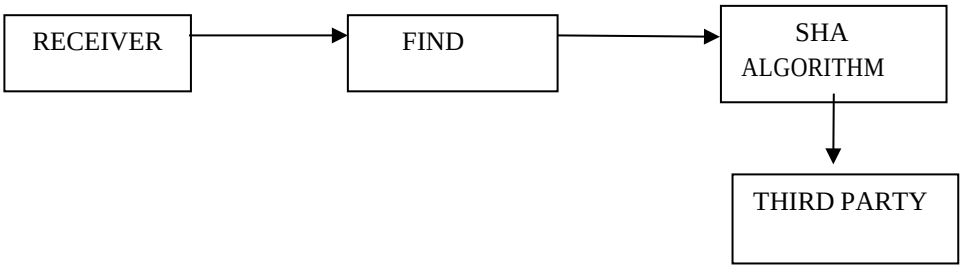


Fig 4.6 Find Third-Party Module

Data Flow Diagram

A Data Flow Diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. Often, they are a preliminary step used to create an overview of the system which can later be elaborated. DFD's can also be used for the visualization of data processing (structured design).

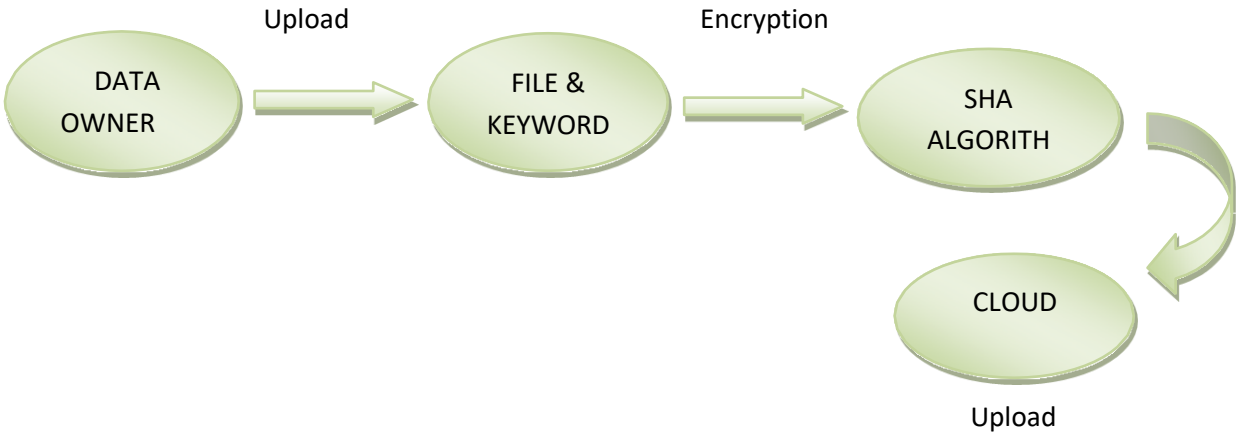


Fig 4.7 DFD-Level 0: Data Owner

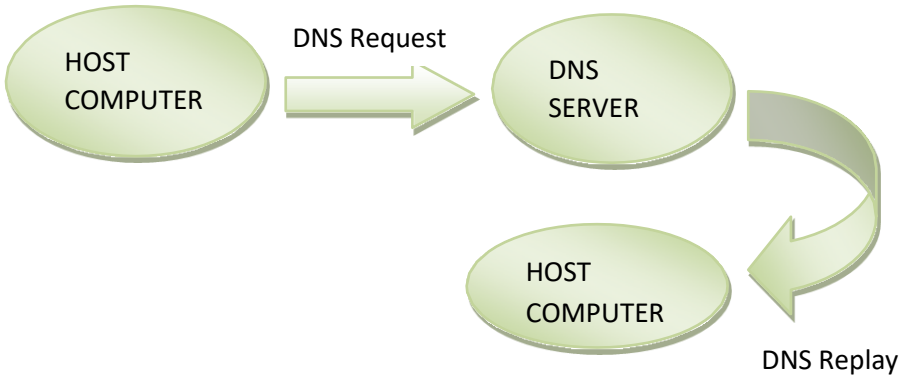


Fig 4.8 DFD-Level 1

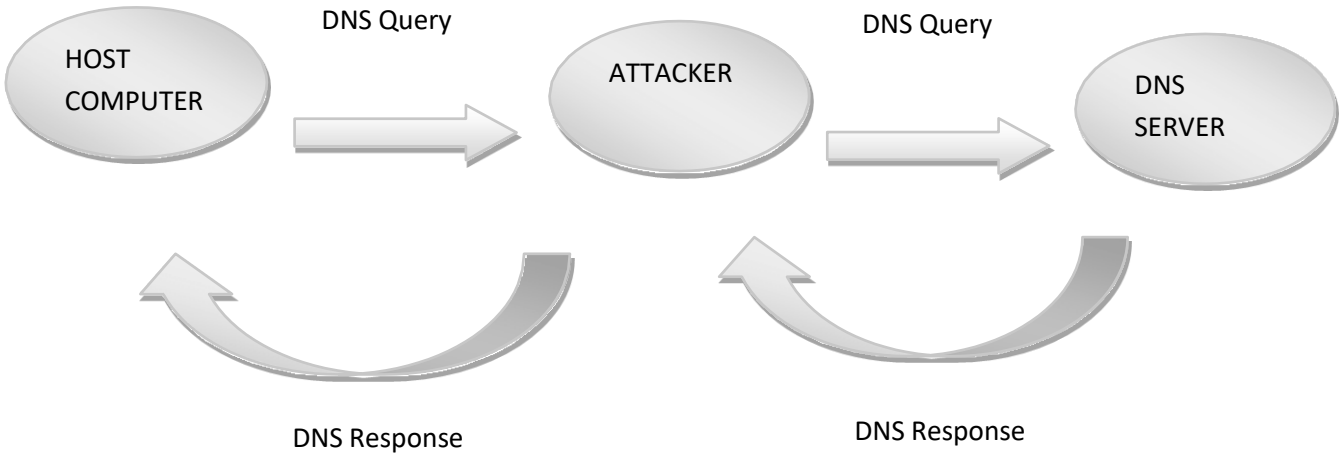


Fig 4.9 DFD-Level 2

PROJECT MANAGEMENT PLAN

Introduction	September 1-30
Literature Survey	October 1-31
System Design	November 1-30
System Implementation	December 1-31
Testing	January 1-30

CHAPTER 5

IMPLEMENTATION DETAILS

DEVELOPMENT AND DEPLOYMENT SETUP

HOW TO INSTALL NETBEANS JAVA IDE ON WINDOWS: Below is a step-by-step process on how to download and install NETBEANS JAVA IDE on Windows:

Step1. To download and install NETBEANS, visit the official website or the setup you can download from the following link:

https://netbeans.org/images_www/v6/download/community/8.2

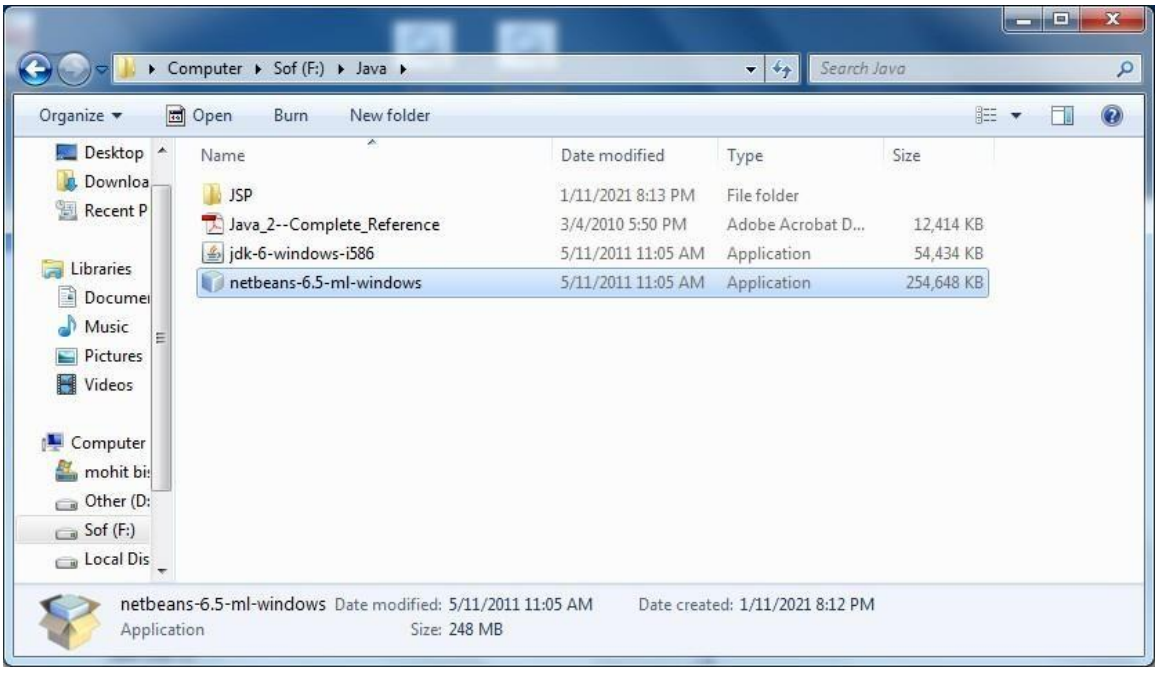


Fig 5.1: A setup file of the NetBeans JAVA

Step 2. Once the download is completed, run the .exe file to install NetBeans. Now click on Install Now. You can download any type of setup as per your requirements from the above mention web page. Right-click on the setup or you can Double-Click on the setup by using the mouse.

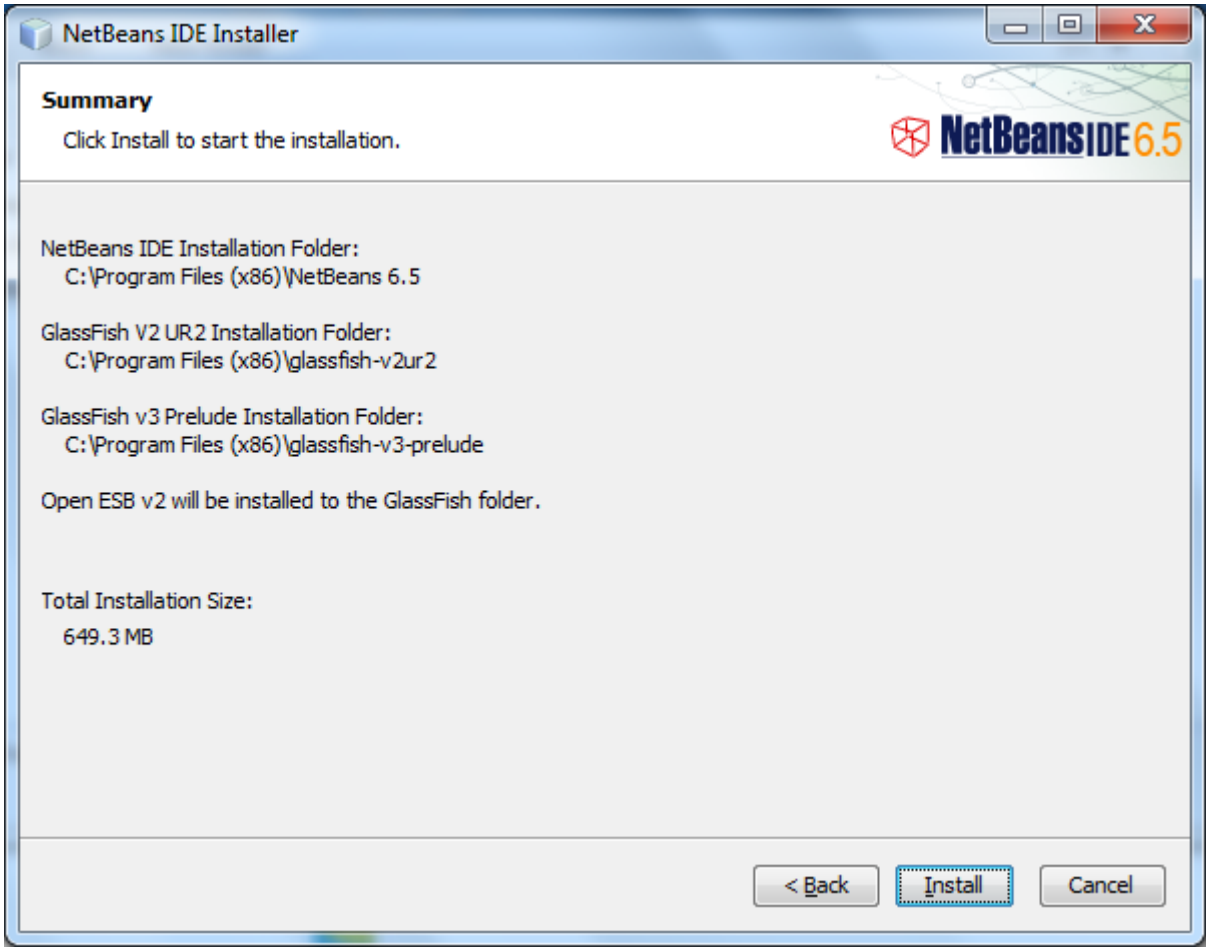


Fig 5.2: Run the .exe file window

Step 3. You can see NetBeans installing at this point.

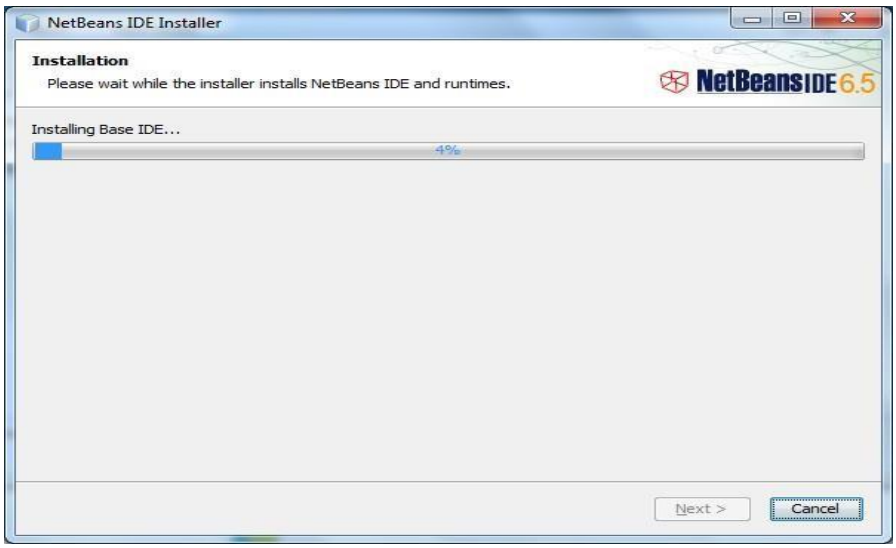


Fig 5.3: NetBeans installing window

Step 4. When it finishes, you can see a screen that says the Setup was successful. Now click on “Close”.

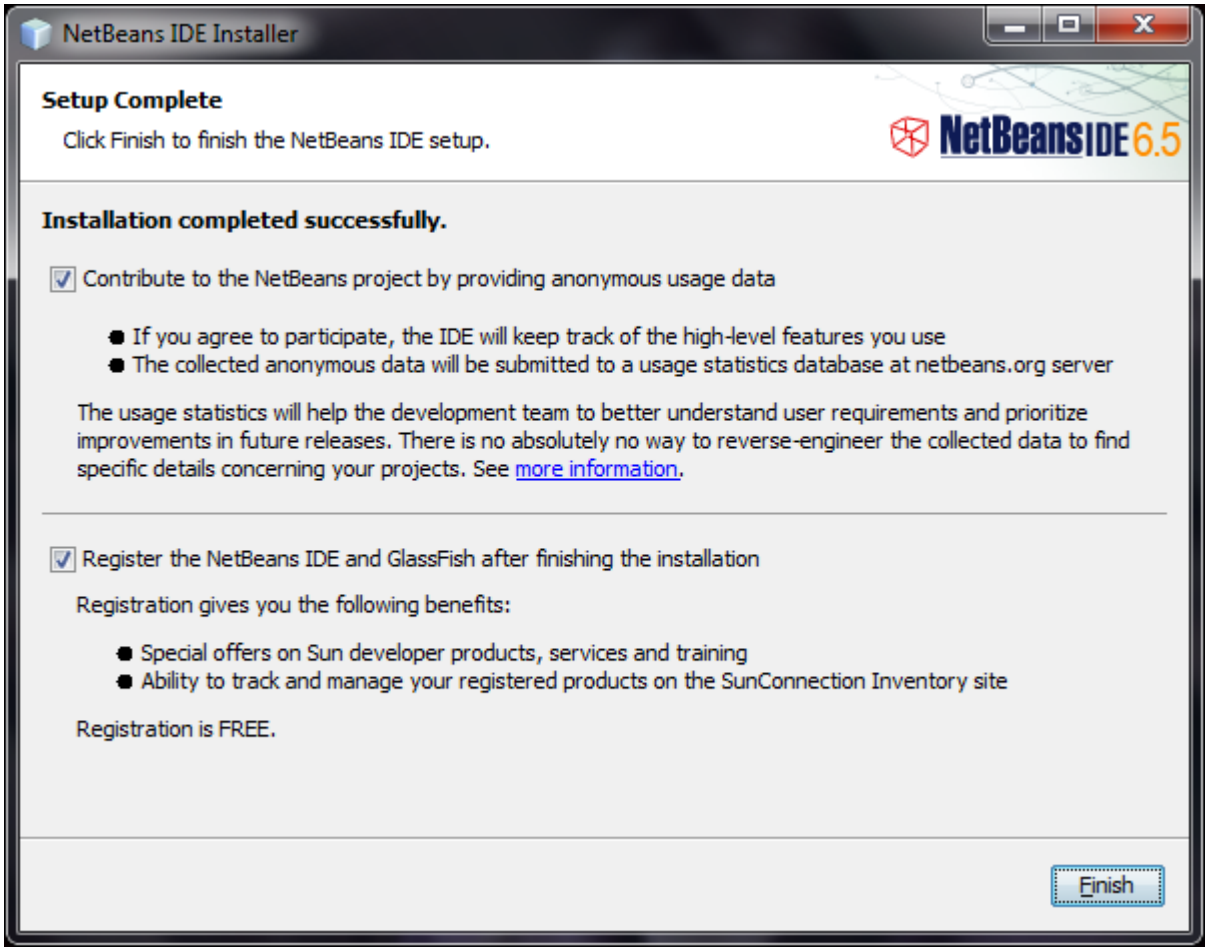


Fig 5.4: NetBeans setup window

HOW TO INSTALL SQL: Here is a step-by-step process on how to download and install SQL on Windows:

Step 1. For our SQL Server Installation steps, we will be first downloading the Developer edition.
Click on Download now and the downloading of will start

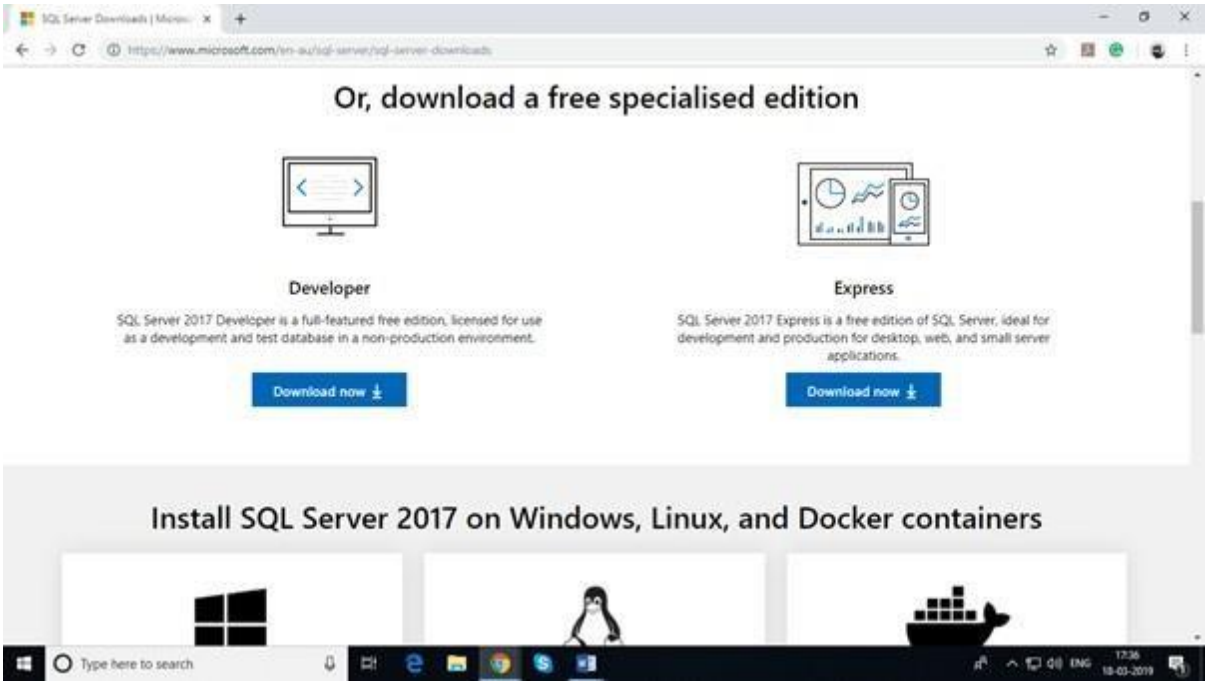


Fig 5.5: SQL Setup website window

Step 2. Click on the file SQLServer2017-SSCI-Dev, and you will find a window with three options: Basic, Custom, and Download Media.

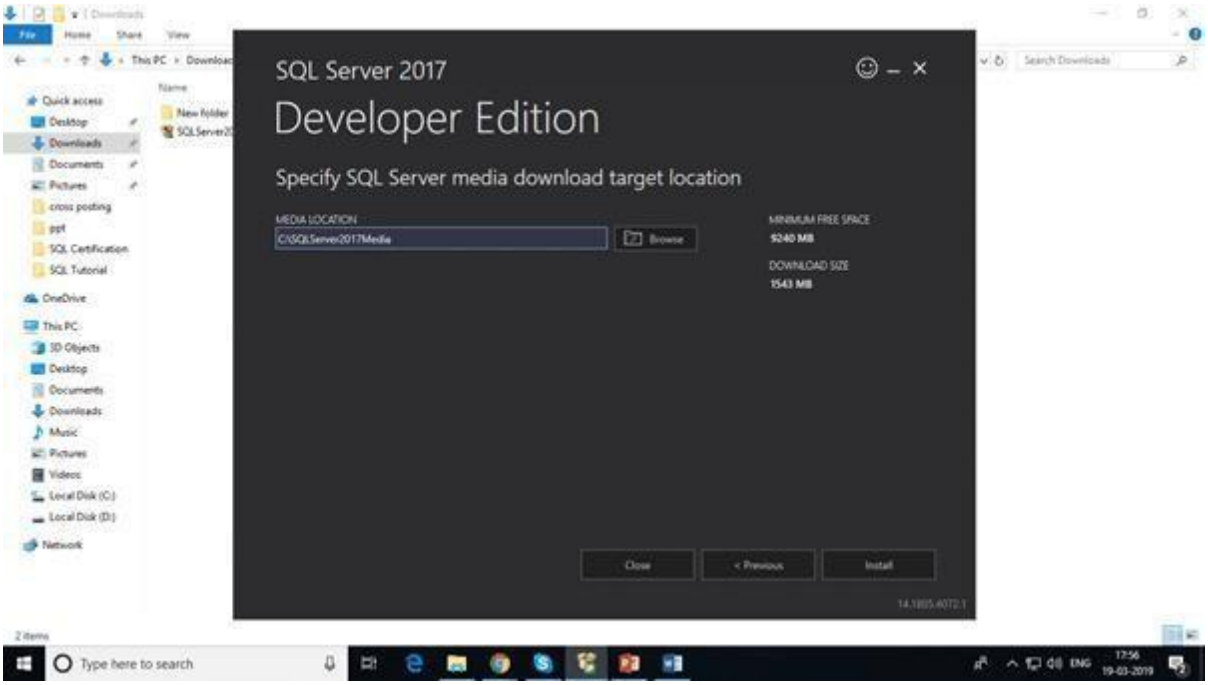


Fig 5.6: SQL Server run the .exe file window

Step 3. On the next screen, Change the installation path if required. Click “Next”.

- Click on the file and install the studio
- Once the installation is completed, search for **Microsoft SQL Server Management Studio**
- The SQL Server Management Studio is launched, and you will find a pop up, **Connect to Server**
- Enter the password which you had entered before with **Login** as **sa** and click on **Connect**

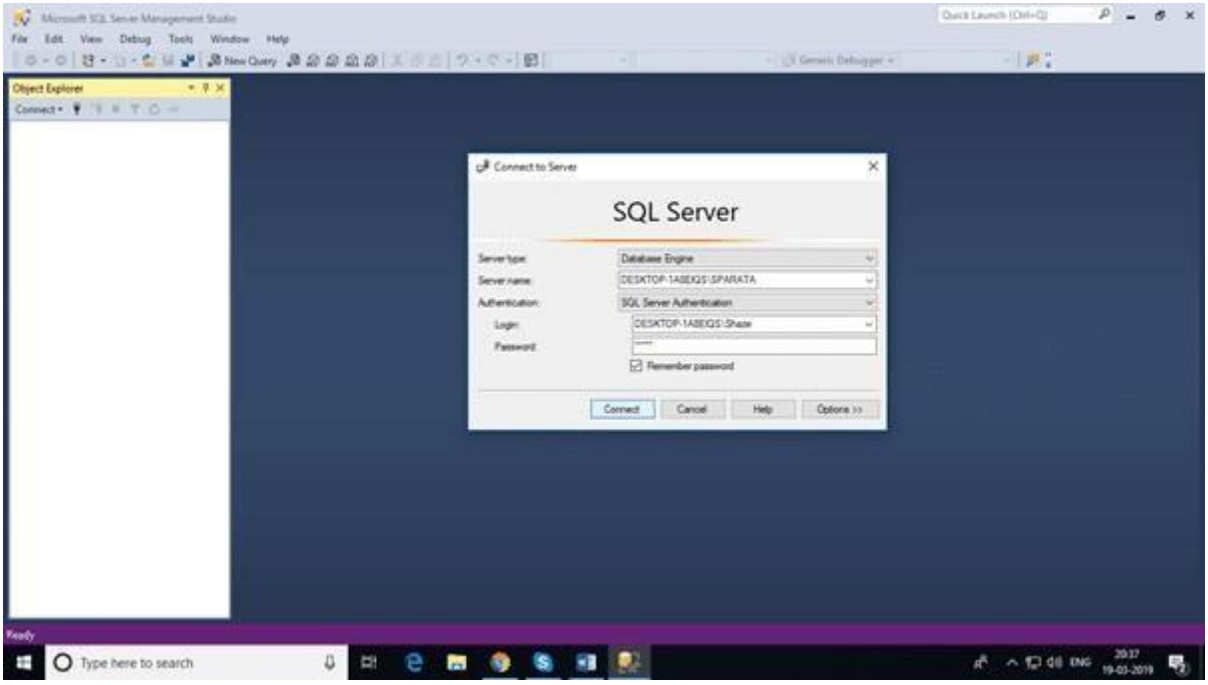


Fig 5.7: SQL setup window

ALGORITHMS

The proposed method for design and implementation of trust-based access control model for cloud computing provides a simpler and faster approach compared to existing methods. The SHA algorithm, or Secure Hash Algorithm, is a widely-usedcryptographic hash function. A cryptographic hash function is a mathematical functionthat takes in a message or data of arbitrary size and produces a fixed-length output, called a hash or digest. The output of a hash function is typically a string of hexadecimaldigits.

The SHA algorithm was developed by the National Security Agency (NSA) in the United States and is now a U.S. Federal Information Processing Standard (FIPS). There are several versions of the SHA algorithm, with varying output sizes and levels of security.

SHA-1 is the first version of the algorithm and produces a 160-bit hash. It was widely used for many years but has been deprecated due to security weaknesses. SHA-2 is the current standard and includes several variations that produce hash lengths of 224, 256, 384, or 512 bits. SHA-3 is the most recent version and was designed as a replacement for SHA-2, although it has not yet seen widespread adoption.

The SHA algorithm is used for a variety of purposes in computer security, including digital signatures, message authentication codes (MACs), and password hashing. When used for digital signatures, the hash of a message is signed with a private key and then verified with the corresponding public key. This ensures the integrity and authenticity of the message. MACs use a shared secret key to generate a hash of the message that can be used to verify that the message has not been tampered with. Password hashing uses the SHA algorithm to convert a plaintext password into a hash that can be stored securely, and then compares the hash of a user's entered password with the stored hash to verify their identity.

The SHA algorithm works by repeatedly applying a compression function to blocks of the message. The compression function takes as input a fixed-size block of the message and the current state of the hash computation, and produces a new state. The output of the compression function is then combined with the previous state to produce the next state. This process is repeated until the entire message has been processed, at which point the final state is the hash of the message.

The security of the SHA algorithm relies on several properties, including collision resistance, preimage resistance, and second preimage resistance. Collision resistance means that it is computationally infeasible to find two different messages that produce the same hash. Preimage resistance means that it is computationally infeasible to find a message that produces a given hash. Second preimage resistance means that it is computationally infeasible to find a second message that produces the same hash as a given message.

Despite its widespread use and proven security, there have been some concerns raised about the potential vulnerability of the SHA algorithm to attacks by quantum computers. This is because the SHA algorithm, like many cryptographic algorithms, relies on the computational difficulty of certain mathematical problems. Quantum computers are capable of solving some of these problems much faster than classical computers, which could potentially weaken the security of the SHA algorithm. However, research is ongoing into post-quantum cryptography, which seeks to develop cryptographic algorithms that are resistant to attacks by both classical and quantum computers.

In conclusion, the SHA algorithm is a widely-used cryptographic hash function that is used for a variety of purposes in computer security. It works by repeatedly applying a compression function to blocks of the message, producing a fixed-length hash that is used to verify the integrity and authenticity of the message. The security of the SHA algorithm relies on several properties, including collision resistance, preimage resistance, and second preimage resistance. While concerns have been raised about the potential vulnerability of the SHA algorithm to attacks by quantum computers, research is ongoing into post-quantum cryptography to develop algorithms that are resistant to such attacks.

The Advanced Encryption Standard (AES) is a widely used symmetric encryption algorithm that is used to protect sensitive information in various applications. AES is a block cipher algorithm that encrypts and decrypts data in blocks of fixed size. It is considered to be one of the most secure encryption algorithms available today.

AES was developed by two Belgian cryptographers, Joan Daemen and Vincent Rijmen, and was chosen by the National Institute of Standards and Technology (NIST) as the successor to the Data Encryption Standard (DES) in 2001. AES uses a substitution-permutation network (SPN) to encrypt and decrypt data. AES operates on blocks of 128 bits, which can be divided into four columns of 32 bits each. The algorithm consists of a series of rounds, with the number of rounds depending on the key size. AES can use key sizes of 128, 192, or 256 bits, with 10, 12, or 14 rounds respectively.

The AES algorithm has four basic operations: SubBytes, ShiftRows, MixColumns, and AddRoundKey. These operations are performed on the state, which is a 4x4 matrix of

bytes that represents the input data. The state is modified in each round of the encryption process.

The SubBytes operation substitutes each byte in the state with a corresponding byte from a fixed substitution table called the S-box. The S-box is generated using a mathematical algorithm and is designed to provide confusion in the encryption process.

The ShiftRows operation shifts the rows of the state cyclically by a certain number of bytes. The first row is not shifted, the second row is shifted one byte to the left, the third row is shifted two bytes to the left, and the fourth row is shifted three bytes to the left. This operation provides diffusion in the encryption process.

The MixColumns operation combines each column of the state with a fixed matrix of coefficients. This operation provides diffusion in the encryption process and helps to prevent patterns from being preserved in the encrypted data.

The AddRoundKey operation combines each byte of the state with a corresponding byte from the round key, which is generated from the main key using a key schedule. The key schedule expands the main key into a series of round keys, one for each round of the encryption process.

In the encryption process, each round consists of these four basic operations, with the exception of the MixColumns operation in the last round. In the decryption process, the operations are performed in reverse order, with the exception of the MixColumns operation, which is replaced by an inverse MixColumns operation.

AES provides a high level of security and is widely used in various applications, including online banking, secure communications, and data storage. It is considered to be one of the most secure encryption algorithms available today and is widely used by governments and security organizations around the world.

However, like any cryptographic algorithm, AES is not infallible and can be susceptible to attacks. There have been various attacks proposed against AES, including differential and linear cryptanalysis, but these attacks require a large amount of computational power and are not practical for most attackers.

In conclusion, AES is a widely used symmetric encryption algorithm that provides a high level of security and is used to protect sensitive information in various applications. It operates on blocks of 128 bits and uses a substitution-permutation network to encrypt

and decrypt data. AES is considered to be one of the most secure encryption algorithms available today and is widely used by governments and security organizations around the world. However, like any cryptographic algorithm, AES is not infallible and can be susceptible to attacks, but these attacks are not practical for most attackers.

The MD5 algorithm is a widely used cryptographic hash function that is used to provide data integrity and to verify the authenticity of digital files. It is often used in computer security applications, such as digital signatures, password storage, and file verification. MD5 was developed in 1991 by Ron Rivest, who also developed the popular RSA encryption algorithm. It was designed to be a fast and efficient algorithm that could be used to generate a fixed-size message digest from variable-length input data.

MD5 takes an input message of any length and produces a fixed-size, 128-bit message digest. The algorithm operates on 512-bit blocks of the input message and uses a series of bitwise logical operations, such as AND, OR, and XOR, as well as modular arithmetic and bitwise rotations.

The MD5 algorithm has four basic steps:

1. Padding - The input message is padded so that its length is a multiple of 512 bits. The padding consists of a single "1" bit, followed by enough "0" bits to fill the remaining space, and then a 64-bit representation of the original message length.
2. Initialization - The algorithm initializes a 128-bit buffer with four fixed values. This buffer is used to store intermediate hash values during the computation.
3. Processing - The algorithm processes the padded message in 512-bit blocks. Each block is processed through a series of four rounds, with each round consisting of sixteen operations.
4. Output - After all blocks have been processed, the 128-bit buffer contains the final hash value, which is the message digest.

The MD5 algorithm has several properties that make it useful for cryptographic applications. One important property is that it is one-way, meaning that it is computationally infeasible to determine the input message from the message digest. This property makes MD5 useful for password storage, as it allows passwords to be stored as message digests rather than as plaintext, making it more difficult for an attacker to obtain the actual passwords.

Another important property of MD5 is that it is collision-resistant, meaning that it is difficult to find two different messages that have the same message digest. This property is important for digital signatures, as it ensures that a digital signature cannot be forged by creating a different message with the same message digest.

Despite its widespread use, the MD5 algorithm is no longer considered to be secure for cryptographic applications. In 2004, a team of researchers demonstrated a successful collision attack against the MD5 algorithm, showing that it is possible to generate two different messages with the same message digest. Since then, several other attacks have been developed that further weaken the security of MD5.

In conclusion, the MD5 algorithm is a widely used cryptographic hash function that is used to provide data integrity and to verify the authenticity of digital files. It operates on 512-bit blocks of input data and produces a fixed-size, 128-bit message digest. MD5 has several important properties, including one-way ness and collision resistance, that make it useful for cryptographic applications. However, due to several successful attacks against the MD5 algorithm, it is no longer considered to be secure for cryptographic applications and has been replaced by newer hash functions, such as SHA-2 and SHA-3.

TEST PROCEDURE / SYSTEM TESTING: Testing is performed to identify errors. It is used for quality assurance. Testing is an integral part of the entire development and maintenance process. The goal of the testing during phase is to verify that the specification has been accurately and completely incorporated into the design, as well as to ensure the correctness of the design itself. For example, the design must not have any logic faults in the design is detected before coding commences, otherwise the cost of fixing the faults will be considerably higher as reflected. Detection of design faults can be achieved by means of inspection as well as walkthrough.

Testing is one of the important steps in the software development phase. Testing checks for the errors, as a whole of the project testing involves the following test cases:

- Static analysis is used to investigate the structural properties of the Source code.
- Dynamic testing is used to investigate the behavior of the source code by executing the program on the test data.

TEST DATA: Unit testing is conducted to verify the functional performance of each modular component of the software. Unit testing focuses on the smallest unit of the software design (i.e.), the module. The white-box testing techniques were heavily employed for unit testing.

FUNCTIONAL TESTS: Functional test cases involved exercising the code with nominal input values for which the expected results are known, as well as boundary values and special values, such as logically related inputs, files of identical elements, and empty files. Three types of tests in Functional test:

- Performance Test
- Stress Test
- Structure Test

INTEGRATION TESTING: Integration testing is a systematic technique for construction the program structure while at the same time conducting tests to uncover errors associated with interfacing. i.e., integration testing is the complete testing of the set of modules which makes up the product. The objective is to take untested modules and build a program structure tester should identify critical modules. Critical modules should be tested as early as possible.

One approach is to wait until all the units have passed testing, and then combine them and then tested. This approach is evolved from unstructured testing of small programs. Another strategy is to construct the product in increments of tested units. A small set of modules are integrated together and tested, to which another module is added and tested in combination. And so on. The advantages of this approach are that, interface dispenses can be easily found and corrected.

The major error that was faced during the project is linking error. When all the modules are combined the link is not set properly with all support files. Then we checked out for interconnection and the links. Errors are localized to the new module and its intercommunications. The product development can be staged, and modules integrated in as they complete unit testing. Testing is completed when the last module is integrated and tested.

TESTING

TESTING TECHNIQUES / TESTING STRATERGIES

Test plan: Software testing is the process of evaluation a software item to detect differences between given input and expected output. Also, to assess the feature of a software item. Testing assesses the quality of the product. Software testing is a process that should be done during the development process. In other words, software testing is a verification and validation process.

Verification: Verification is the process to make sure the product satisfies the conditions imposed at the start of the development phase. In other words, to make sure the product behaves the way we want it to.

Validation: Validation is the process to make sure the product satisfies the specified requirements at the end of the development phase. In other words, to make sure the product is built as per customer requirements.

Basics of software testing: There are two basics of software testing: black box testing and white box testing.

Black box Testing: Black box testing is a testing technique that ignores the internal mechanism of the system and focuses on the output generated against any input and execution of the system. It is also called functional testing.

White box Testing: White box testing is a testing technique that takes into account the internal mechanism of a system. It is also called structural testing and glass box testing. Black box testing is often used for validation and white box testing is often used for verification.

Types of testing: There are many types of testing like

- Unit Testing
- Integration Testing
- Functional Testing
- System Testing
- Stress Testing
- Performance Testing
- Usability Testing
- Acceptance Testing

- Regression Testing
- Beta Testing

Unit Testing: Unit testing is the testing of an individual unit or group of related units. It falls under the class of white box testing. It is often done by the programmer to test that the unit he/she has implemented is producing expected output against given input.

Integration Testing: Integration testing is testing in which a group of components are combined to produce output. Also, the interaction between software and hardware is tested in integration testing if software and hardware components have any relation. It may fall under both white box testing and black box testing.

Functional Testing: Functional testing is the testing to ensure that the specified functionality required in the system requirements works. It falls under the class of black box testing.

System Testing: System testing is the testing to ensure that by putting the software in different environments (e.g., Operating Systems) it still works. System testing is done with full system implementation and environment. It falls under the class of black box testing.

Stress Testing: Stress testing is the testing to evaluate how system behaves under unfavorable conditions. Testing is conducted at beyond limits of the specifications. It falls under the class of black box testing.

Performance Testing: Performance testing is the testing to assess the speed and effectiveness of the system and to make sure it is generating results within a specified time as in performance requirements. It falls under the class of black box testing.

Usability Testing: Usability testing is performed to the perspective of the client, to evaluate how the GUI is user-friendly? How easily can the client learn? After learning how to use, how proficiently can the client perform? How pleasing is it to use its design? This falls under the class of black box testing.

Acceptance Testing: Acceptance testing is often done by the customer to ensure that the delivered product meets the requirements and works as the customer expected. It falls under the class of black box testing.

Regression Testing: Regression testing is the testing after modification of a system, component, or a group of related units to ensure that the modification is working

correctly and is not damaging or imposing other modules to produce unexpected results. It falls under the class of black box testing.

Requirement Analysis: Requirement analysis, also called requirement engineering, is the process of determining user expectations for a new modified product. It encompasses the tasks that determine the need for analysing, documenting, validating and managing software or system requirements. The requirements should be documentable, actionable, measurable, testable and traceable related to identified business needs or opportunities and define to a level of detail, sufficient for system design.

FUNCTIONAL REQUIREMENTS

It is a technical specification requirement for the software products. It is the first step in the requirement analysis process which lists the requirements of particular software systems including functional, performance and security requirements. The function of the system depends mainly on the quality hardware used to run the software with given functionality.

Usability: It specifies how easy the system must be use. It is easy to ask queries in any format which is short or long, porter stemming algorithm stimulates the desired response for user.

Robustness: It refers to a program that performs well not only under ordinary conditions but also under unusual conditions. It is the ability of the user to cope with errors for irrelevant queries during execution.

Security: The state of providing protected access to resource is security. The system provides good security and unauthorized users cannot access the system there by providing high security.

Reliability: It is the probability of how often the software fails. The measurement is often expressed in MTBF (Mean Time Between Failures). The requirement is needed in order to ensure that the processes work correctly and completely without being aborted. It can handle any load and survive and survive and even capable of working around any failure.

Compatibility: It is supported by version above all web browsers. Using any web servers like localhost makes the system real-time experience.

Flexibility: The flexibility of the project is provided in such a way that is has the ability to run on different environments being executed by different users.

Safety: Safety is a measure taken to prevent trouble. Every query is processed in a secured manner without letting others to know one's personal information.

NON- FUNCTIONAL REQUIREMENTS

Portability: It is the usability of the same software in different environments. The project can be run in any operating system.

Performance: These requirements determine the resources required, time interval, throughput and everything that deals with the performance of the system.

Accuracy: The result of the requesting query is very accurate and high speed of retrieving information. The degree of security provided by the system is high and effective.

Maintainability: Project is simple as further updates can be easily done without affecting its stability. Maintainability basically defines that how easy it is to maintain the system. It means that how easy it is to maintain the system, analyses, change and test the application. Maintainability of this project is simple as further updates can be easily done without affecting its stability.

CHAPTER 6
RESULTS AND DISCUSSION

Through the above summary, due to the problems about the collusion attacks that are widespread in the secure outsourcing of sequence comparison algorithms, this paper will introduce the trusted authority to authenticate user those who have the access to the data on cloud. SHA algorithm is used by the trusted authority to generate the key and that key will get share to user as well as the owner. The trusted authority module receives encrypted file using AES Algorithm from the data owner and computes hash value using MD-5 algorithm. It stores key in its database which will be used during the dynamic operations and to determine the cheating party in the system. Trustedauthority send file to CSP module to store on cloud. The resulting key sets are shown to have a number of desirable properties that ensure the confidentiality of communication sessions against collusion attacks by other network nodes.

Trust-based access control is a model used to control access to resources in a cloud computing environment based on the level of trust established between the entities involved. This model involves evaluating the trustworthiness of the user or application requesting access to a resource and making a decision on whether to grant or deny access based on that evaluation.

The trust evaluation is based on several factors, including the user's identity, past behavior, and reputation, as well as the context in which the access request is made. This context includes information about the resource being accessed, the sensitivity of the data being accessed, and the location and device used to make the request. One of the main benefits of the trust-based access control model is that it provides a more granular and context-aware approach to access control. This means that access can be more finely tuned based on the specific circumstances surrounding each request.

However, there are also some challenges to implementing a trust-based access control model. One of the biggest challenges is establishing a framework for evaluating trust that is both accurate and reliable. This requires collecting and analyzing large amounts of data on user behavior and reputation, as well as developing sophisticated algorithms for processing this data.

Another challenge is ensuring that the access control system is scalable and can handle the large volumes of requests that are typical in a cloud computing environment. This requires careful design and optimization of the system architecture and algorithms.

Overall, the trust-based access control model is an important development in cloud computing security, providing a more nuanced and flexible approach to access control that can adapt to the changing needs of cloud environments. However, implementing this model requires careful consideration of the various challenges and trade-offs involved.

Cloud computing has become an essential part of modern-day computing, offering users the ability to access computing resources and services over the internet. With the rise of cloud computing, security has become a significant concern. One of the primary security concerns is access control, which ensures that only authorized users or applications can access resources and data in the cloud. Traditional access control models such as role-based access control (RBAC) and attribute-based access control (ABAC) are widely used in cloud environments.

However, these models have limitations in terms of providing granular and context-aware access control. To overcome these limitations, trust-based access control (TBAC) models have been proposed. It is a model used to control access to resources in a cloud computing environment based on the level of trust established between the entities involved. TBAC evaluates the trustworthiness of the user or application requesting access to a resource and makes a decision on whether to grant or deny access based on that evaluation. The trust evaluation is based on several factors, including the user's identity, past behavior, and reputation, as well as the context in which the access request is made. TBAC is a more context-aware and granular approach to access control than traditional models such as RBAC and ABAC.

In RBAC, access control decisions are based on a user's assigned role, which may not take into account the specific context in which the access request is made. ABAC provides more fine-grained access control by allowing policies to be defined based on attributes such as time, location, and device. However, ABAC still relies on predefined policies and does not adapt to the changing trustworthiness of the user or application. TBAC, on the other hand, uses a dynamic trust evaluation process to

determine access control decisions based on the specific context of each access request. This allows TBAC to adapt to the changing trustworthiness of users or applications and provide more flexible access control.

Trust Evaluation Factors:

The trust evaluation in TBAC is based on several factors. These factors include:

Identity: The user's identity is a crucial factor in the trust evaluation process. TBAC considers the user's identity in determining whether to grant or deny access to a resource.

Past Behavior: TBAC considers the user's past behavior when evaluating trust. For example, if a user has a history of accessing sensitive data without authorization, TBAC may assign a lower trust score to that user.

Reputation: TBAC also considers the user's reputation when evaluating trust. This may include information about the user's previous interactions with other users or applications in the cloud environment.

Context: The context in which the access request is made is a crucial factor in the trust evaluation process. This includes information about the resource being accessed, the sensitivity of the data being accessed, and the location and device used to make the request. Third-Party Trust: TBAC may also consider third-party trust evaluations when making access control decisions. For example, if a user has been previously evaluated and trusted by a trusted third-party service, TBAC may assign a higher trust score to that user.

The trust evaluation process in TBAC involves several steps:

Collection of Data: TBAC collects data about the user or application requesting access to a resource. This may include information about the user's identity, past behavior, reputation, and context. TBAC uses a trust evaluation algorithm to process the data collected and assign a trust score to the user or application. The trust evaluation algorithm takes into account the various factors described above and may use machine learning or other advanced techniques.

CHAPTER 7
CONCLUSION

CONCLUSION

According to the above summary, a trusted authority to verify a user who has access to cloud data is necessary owing to the pervasive issue of collusion assaults in the secure outsourcing of sequence comparison algorithms. The key is created by the trustworthy authority using the SHA algorithm and is then sent to both the user and the owner. The trusted authority module computes the hash value using the MD-5 method after receiving an AES-encrypted file from the data owner. Its database includes keys that may be used to distinguish the system's third party during dynamic operations. The answer to access control in cloud computing settings is a trust-based access control approach. It provides more fine-grained and dynamic access control mechanisms that take into account the trust level of users and resources. By integrating trust metrics, trust policies, and trust evaluation processes, access control can improve security, increase flexibility, and reduce management complexity. However, the implementation of a trust-based access control model must be carefully designed and evaluated to ensure its effectiveness and efficiency. As well as concerns around the privacy and security of data, must be carefully considered. Ongoing monitoring are also essential to ensure that the access control model is working as intended and to identify any potential issues or areas for improvement. Overall, a trust-based access control model can help organizations to better manage and control access to their cloud resources while also enhancing the user experience by providing more flexible and efficient access control mechanisms. As cloud computing continues to evolve and become more prevalent, the development and implementation of effective access models will be critical to ensuring the security and reliability of cloud services.

FUTURE WORK

Future research in It is somewhat hard to extend the work in our paper to certain applications with multi-data source. Firstly, two-character sequences from different sources should be encrypted respectively with different keys. Secondly, three cost matrices should be encrypted together after being constructed by the negotiation

between both sides. The security target is to complete sequence comparison on a single cloud server in the way of privacy preservation and to ensure that the string typed data of the end user on any side will not be arbitrarily stolen by the other user or the CSP.

RESEARCH ISSUES

The design and implementation of a trust-based access control model for cloud computing poses several research issues that need to be addressed in order to ensure that the model is effective and secure. Here are some of the key research issues that need to be considered:

1. Trust management: A trust-based access control model relies on accurate and reliable trust information to make access decisions. This requires the development of effective mechanisms for trust management, including trust evaluation, trust propagation, and trust update. Research needs to focus on developing efficient and effective trust management mechanisms that can accurately reflect the trustworthiness of cloud service providers.
2. Security and privacy: The trust-based access control model needs to ensure that the security and privacy of user data is protected at all times. This requires the implementation of appropriate security and privacy mechanisms, such as encryption and data anonymization, as well as effective access control policies that are designed to prevent unauthorized access or disclosure of user data.
3. Scalability: Cloud computing environments can be extremely large and complex, and the trust-based access control model needs to be scalable enough to accommodate the large number of users, devices, and services that may be involved. Research needs to focus on developing scalable trust management mechanisms that can handle large volumes of data and transactions without compromising performance or security.
4. Interoperability: The trust-based access control model needs to be interoperable with existing cloud computing environments, including different cloud service providers, cloud platforms, and access control systems. This requires the development of standards and protocols that can ensure interoperability and facilitate the integration of different components within the cloud computing ecosystem.

5. User acceptance: The success of a trust-based access control model depends on user acceptance and adoption. Research needs to focus on developing user-friendly interfaces and mechanisms that can help users understand the trustworthiness of cloud service providers and make informed decisions about access control.

6. Accountability: The trust-based access control model needs to ensure that cloud service providers are held accountable for their actions and that they are transparent about their security and privacy practices. This requires the development of effective mechanisms for auditing and monitoring cloud service providers, as well as mechanisms for enforcing compliance with security and privacy policies.

In conclusion, the design and implementation of a trust-based access control model for cloud computing is a complex and challenging task that requires addressing several research issues related to trust management, security and privacy, scalability, interoperability, user acceptance, and accountability. Addressing these issues will help to ensure that the trust-based access control model is effective and secure, and that it can provide users with the necessary trust information to make informed decisions about access control in cloud computing environments.

IMPLEMENTATION ISSUES

The implementation of a trust-based access control model for cloud computing involves several implementation issues that need to be considered in order to ensure that the model is effectively and efficiently implemented. Here are some of the key implementation issues that need to be addressed:

1. Implementation architecture: The design and implementation of a trust-based access control model requires careful consideration of the architecture that will be used to implement the model. This includes the selection of appropriate hardware and software platforms, as well as the identification of the necessary components and modules that will be required to implement the model.

2. Trust evaluation: The implementation of a trust-based access control model requires the development of mechanisms for evaluating the trustworthiness of cloud service providers. This includes the development of trust models, trust evaluation algorithms, and trust scoring mechanisms that can be used to assess the trustworthiness of cloud service providers.

3. Access control policy: The implementation of a trust-based access control model requires the development of access control policies that can be used to enforce access decisions based on the trustworthiness of cloud service providers. This requires the identification of the necessary access control rules and the development of the necessary mechanisms for enforcing those rules.

4. Privacy and security mechanisms: The implementation of a trust-based access control model requires the implementation of appropriate privacy and security mechanisms that can protect the confidentiality, integrity, and availability of user data. This includes the implementation of encryption, authentication, and access control mechanisms that can prevent unauthorized access or disclosure of user data.

5. Integration with existing systems: The implementation of a trust-based access control model needs to be integrated with existing cloud computing environments, including different cloud service providers, cloud platforms, and access control systems. This requires the development of standards and protocols that can ensure interoperability and facilitate the integration of different components within the cloud computing ecosystem.

6. User training and awareness: The implementation of a trust-based access control model requires user training and awareness programs to ensure that users understand the trustworthiness of cloud service providers and how to make informed decisions about access control. This requires the development of user-friendly interfaces and mechanisms that can help users understand the trustworthiness of cloud service providers and make informed decisions about access control.

7. Monitoring and auditing mechanisms: The implementation of a trust-based access control model requires the development of monitoring and auditing mechanisms that can be used to monitor the trustworthiness of cloud service providers and ensure that they are complying with security and privacy policies. This requires the development of effective mechanisms for auditing and monitoring cloud service providers, as well as mechanisms for enforcing compliance with security and privacy policies.

In conclusion, the implementation of a trust-based access control model for cloud computing requires careful consideration of several implementation issues related to the implementation architecture, trust evaluation, access control policy, privacy and

security mechanisms, integration with existing systems, user training and awareness, and monitoring and auditing mechanisms. Addressing these implementation issues will help to ensure that the trust-based access control model is effectively and efficiently implemented and that it can provide users with the necessary trust information to make informed decisions about access control in cloud computing environments.

REFERENCES

1. Chen, L. C., Papandreou, G., Schroff, F., & Adam, H. (2018). Rethinking atrous convolution for semantic image segmentation. arXiv preprint arXiv:1706.05587.
2. Chen, L. C., Zhu, Y., Papandreou, G., Schroff, F., & Adam, H. (2018). Encoder-decoder with atrous separable convolution for semantic image segmentation. In Proceedings of the European conference on computer vision (ECCV) (pp. 801-818).
3. Chen, L. C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2018). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE transactions on pattern analysis and machine intelligence, 40(4), 834-848.
4. Liu, C., Chen, L. C., Schroff, F., Adam, H., Hua, W., Yuille, A., & Fei-Fei, L. (2019). Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 82-92).
5. Zhang, L., Wang, L., Li, Y., & Zhang, L. (2021). DeepLabv3+: Encoder-decoder with Atrous Separable Convolution for Semantic Segmentation. Journal of Computer Science and Technology, 36(4), 655-664.
6. Daniel Donatsch;Nico Färber;Matthias Zwicker ,3D conversion using vanishing points and image warping.
7. Chun-Jen Tsai, & Katsaggelos, A. K. (2000). Sequential construction of 3-D-based scene description.
8. Papadimitriou, D. V. (1995). A stereo disparity algorithm for 3D model construction.
9. Kitayama, D., Touma, Y., Hagiwara, H., Asami, K., & Komori, M. (2015). 3D map construction based on structure from motion using stereo vision.
10. Tao Ni, Yamada, H., & Hongyan Zhang. (2010). Image based real-time 3D reconstruction for teleoperation system.
11. Haoxue, L., Wenjie, Y., Bing, W., & Min, H. (2008). A Method to Get Realistic 3D Sensation Image from a Single Image.
12. Abdel-Aziz, Y.I., Karara, H.M., 1971. Direct linear transformation from computer coordinates into object coordinates in close-range photogrammetry. In:

Proceedings of ASP/UI Symposium of Close-Range Photogrammetry, Illinois,USA, January 1971, pp. 1–18.

13. Ayoub, A.F., Siebert, P., Moos, K.F., Wray, D., Urquhart, C., Niblett, T.B., 1998a.A vision-based three-dimensional capture system for maxillofacial assessment and surgical planning. British Journal of Maxillofacial Surgery 36 (5), 353–357.
14. Ayoub, A.F., Wray, D., Moos, K.F., Siebert, P., Jin, J., Niblett, T.B., Urquhart, C., Mowforth, P., 1998b. Three-dimensional modelling for modern diagnosis and planning in maxillofacial surgery. International Journal of Adult Orthodontics and Orthognathic Surgery 11 (3), 225–233.
15. Brandl, N., Jorgensen, E., 1996. Determination of live weight of pigs from dimensions measured using image analysis. Computers and Electronics in Agriculture 15 (1), 57–72.
16. Chen, Y., Medioni, G., 1992. Object modeling by registration of multiple range images. Image and Vision Computing 10 (3).
17. Coffey, R.D., Parker, G.R., Laurent, K.M., 1999. Assessing sow body condition, ASC-158 Publication on-line at http://www.thepigsite.com/FeaturedArticle/Default.asp?Display=275.
18. Curless, B., Levoy, M., 1996. Volumetric method for building complex models from range images. In: Proceedings of the Computer Graphics Conference, SIGGRAPH 96, New Orleans, USA, 4–9 August 1996, pp. 303–312.
19. DEFRA, 2002. Information booklet on animal welfare: condition scoring of pigs. Publication on-line at http://www.defra.gov.uk/animalh/welfare/farmed/pigs/pb3480/pigsctoc.htm, Department of Environment, Food and Rural Affairs, UK.
20. Faugeras, O., 2001. Three-dimensional Computer Vision. MIT Press, London, pp. 165–243 (Chapter 6).
21. Fearon, P., 2002. DPI pigtech notes: sow condition scoring. Publication on-line at http://www.dpi. qld.gov.au/pigs/4324.html, Department of Primary Industries, Queensland, Australia.

APPENDIX

A. SOURCE CODE:

```
package com.mona;


/**

 *

 * @author sentamilpandi.m

 */

import java.io.*;

import java.security.*;

import java.security.spec.EncodedKeySpec;

import java.security.spec.PKCS8EncodedKeySpec;

import java.security.spec.X509EncodedKeySpec;

import javax.crypto.*;

import org.apache.commons.codec.binary.Hex;

import org.bouncycastle.jce.provider.BouncyCastleProvider;

import sun.misc.BASE64Decoder;

import sun.misc.BASE64Encoder;


public class EDCRYPT {

    String ALGORITHM_USED = "RSA";

    String PROVIDER = "BC";
```

```
private KeyPair key;

public EDCRYPT() throws NoSuchAlgorithmException
{
    this.init();
    this.generateKey();
}

public void init()
{
    Security.addProvider(new BouncyCastleProvider());

}

public KeyPair generateKey() throws NoSuchAlgorithmException
{
    KeyPairGenerator keyGen = null;
    try {
        keyGen = KeyPairGenerator.getInstance(ALGORITHM_USED, PROVIDER);
    }catch (NoSuchProviderException e){e.printStackTrace();}
    keyGen.initialize(1024);
```

```
    key = keyGen.generateKeyPair();
    return key;
}

public PublicKey getpublickey()
{
    return key.getPublic();
}
public PrivateKey getprivatekey()
{
    return key.getPrivate();
}

public byte[] encrypt(byte[] text, PublicKey key) throws Exception
{
    byte[] cipherText = null;
    try
    {
        Cipher cipher = Cipher.getInstance("RSA/ECB/PKCS1Padding",PROVIDER);
        cipher.init(Cipher.ENCRYPT_MODE, key);
        cipherText = cipher.doFinal(text);
```

```
    }catch (Exception e){throw e;}
    return cipherText;
}

public String encrypt(String text, PublicKey key) throws Exception
{
    String encryptedText;
    try
    { byte[] cipherText = encrypt(text.getBytes(),key);
        encryptedText = encodeToBASE64(cipherText);
    }catch (Exception e){throw e;}return encryptedText;
}

public byte[] decrypt(byte[] text, PrivateKey key) throws Exception
{
    byte[] dectyptedText = null;
    try
    {
        Cipher cipher = Cipher.getInstance("RSA/ECB/PKCS1Padding",PROVIDER);
        cipher.init(Cipher.DECRYPT_MODE,key);
        dectyptedText = cipher.doFinal(text);
```

```
    }catch (Exception e){throw e;}
    return dectyptedText;
}

public String decrypt(String text, PrivateKey key) throws Exception
{
    String result;
    try
    {  byte[] dectyptedText = decrypt(decodeToBASE64(text),key);
        result = new String(dectyptedText);
    }catch (Exception e){throw e;}
    return result;
}


public String getKeyAsString(Key key)
{
    byte[] keyBytes = key.getEncoded();
    BASE64Encoder b64 = new BASE64Encoder();
    return b64.encode(keyBytes);
}
```

```
public PrivateKey getPrivateKeyFromString(String key) throws Exception
{
    KeyFactory keyFactory = KeyFactory.getInstance(ALGORITHM_USED);
    BASE64Decoder b64 = new BASE64Decoder();
    EncodedKeySpec privateKeySpec = new
PKCS8EncodedKeySpec(b64.decodeBuffer(key));
    PrivateKey privateKey = keyFactory.generatePrivate(privateKeySpec);
    return privateKey;
}

public PublicKey getPublicKeyFromString(String key) throws Exception
{
    BASE64Decoder b64 = new BASE64Decoder();
    KeyFactory keyFactory = KeyFactory.getInstance(ALGORITHM_USED);
    EncodedKeySpec publicKeySpec = new
X509EncodedKeySpec(b64.decodeBuffer(key));
    PublicKey publicKey = keyFactory.generatePublic(publicKeySpec);
    return publicKey;
}

private String encodeToBASE64(byte[] bytes)
```

```
{
BASE64Encoder b64 = new BASE64Encoder();
return b64.encode(bytes);
}

private byte[] decodeToBASE64(String text) throws IOException
{
    BASE64Decoder b64 = new BASE64Decoder();
    return b64.decodeBuffer(text);
}

public static void main(String[] args) throws Exception {
    EDCRYPT rsa= new EDCRYPT();

    String pub=rsa.getKeyAsString(rsa.getpublickey());
    PublicKey key=rsa.getPublicKeyFromString(pub);
    System.out.println("public key ???"+pub);
    String encry=rsa.encrypt("hello world is the first java program for the java beginners",key);
    System.out.println("cipher text :"+encry);
    String pri=rsa.getKeyAsString(rsa.getprivatekey());
    System.out.println("private key "+pri);
```

```
    String decry=rsa.decrypt(encry,rsa.getPrivateKeyFromString(pri));

    System.out.println("!!!!!!!!!"+decry);

}


}

package com.mona;

import com.commondb.Common_DB;
import java.io.File;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import  javax.servlet.RequestDispatcher;
import javax.servlet.http.HttpSession;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.logging.Level;
import java.util.logging.Logger;
import java.util.ArrayList;


/**
 *
 * @author sentamilpandi.m
```

```
 */
public class LoginServlet extends HttpServlet {
    String UserName="";
    String Password="";
    String Requestgroup;
    String filename;
    String Email="";
    String group="";
    Connection con=null;
    Statement st=null;
    ResultSet rs=null;
    RequestDispatcher rd=null;


    /**
     * Processes requests for both HTTP
     * <code>GET</code> and
     * <code>POST</code> methods.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
            throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            /*
```

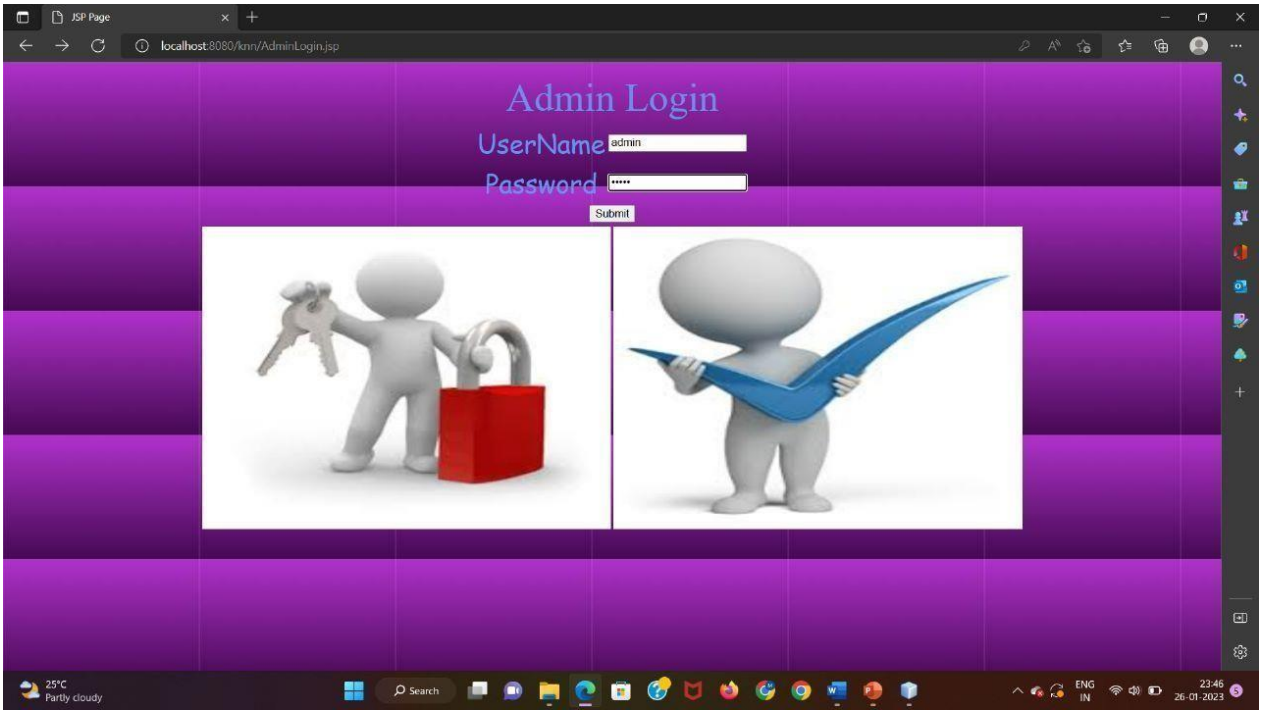
```
            * TODO output your page here. You may use following sample code.
            */
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Servlet LoginServlet</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Servlet LoginServlet at " + request.getContextPath() +
"</h1>");
            out.println("</body>");
            out.println("</html>");
        } finally {
            out.close();
        }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the +
sign on the left to edit the code.">
    /**
     * Handles the HTTP
     * <code>GET</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
```

```
        processRequest(request, response);
    }

    /**
     * Handles the HTTP
     * <code>POST</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        HttpSession session1=request.getSession(true);
        UserName=request.getParameter("UserName");
        Password=request.getParameter("Password");
        Requestgroup=request.getParameter("Requestgroup");
        filename=request.getParameter("filename");
        try {
            ArrayList list=new ArrayList();
            ResultSet rs1=Common_DB.LoginCheck("mona", "Login",
"UserName","Password",UserName, Password);
            if(rs1.next()) {
                String group=rs1.getString("groupname");
                //String group=request.getParameter("groupname");
                System.out.println(">>>>>>>>>>"+group);
                File file=new File("D:/"+group);
              if(!(file.exists()))
                {
```

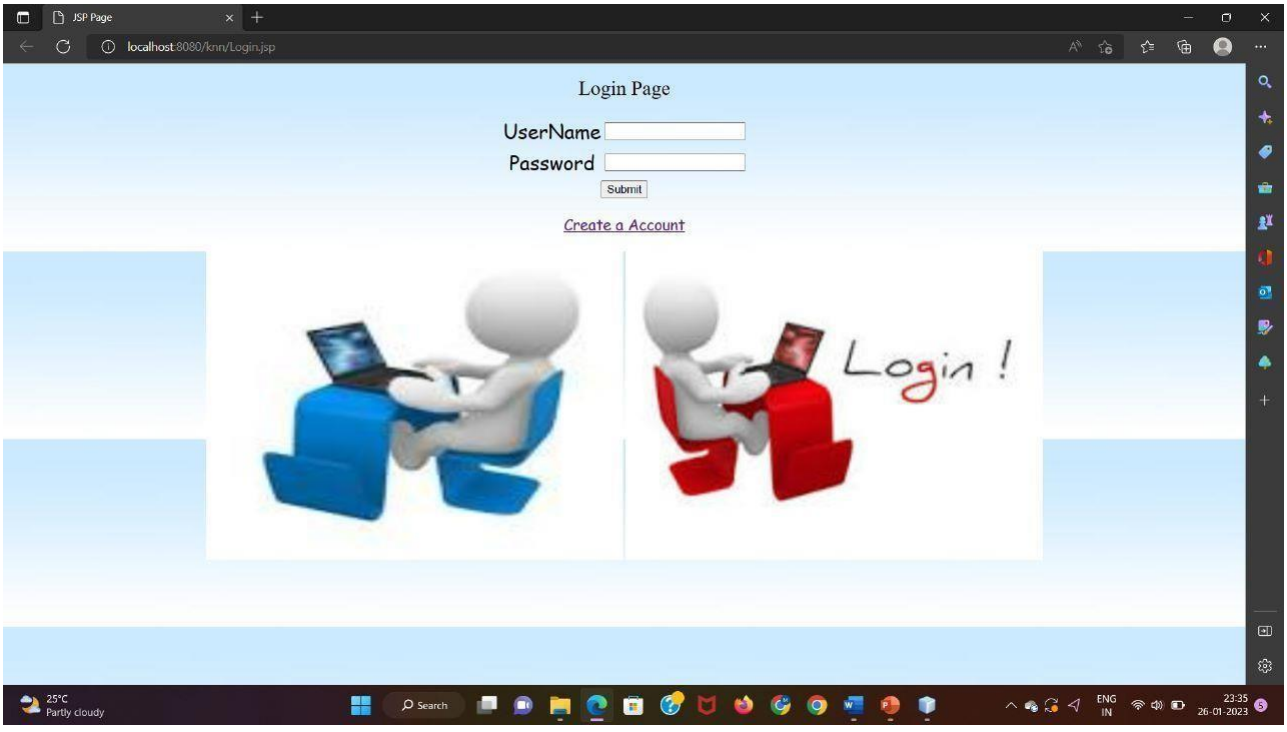
```
                file.mkdir();

              }
              File[] files=new File("D:/"+group).listFiles();
               System.out.println(">>>>>>>>>>"+files.length);
                 //if(files!=null && files.length>0)
                    for(int i=0;i<files.length;i++) {
                       String filename=files[i].getName();
                       list.add(filename);
                    }
            session1.setAttribute("group", group);
            session1.setAttribute("filename", list);
            session1.setAttribute("username", UserName);
            response.sendRedirect("download.jsp");

            Class.forName("com.mysql.jdbc.Driver");

con=DriverManager.getConnection("jdbc:mysql://localhost:3306/mona","root","passw
ord");
            st=con.createStatement();
            rs=st.executeQuery("select * from login where UserName='"+UserName+"'
and Password='"+Password+"' and groupname='"+group+"'");
            if(rs.next())
            {
              System.out.println("COMING");
            response.sendRedirect("download.jsp");
            }
            else
            {
                response.sendRedirect("download1.jsp");
            }
```

```
            }
            else
            {
                response.sendRedirect("Error.jsp");
            }

        }
        catch (Exception ex)
        {
            ex.printStackTrace();
            //Logger.getLogger(LoginServlet.class.getName()).log(Level.SEVERE,      null,
ex);
        }


        //processRequest(request, response);

    }

    /**
     * Returns a short description of the servlet.
     *
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>
}
```

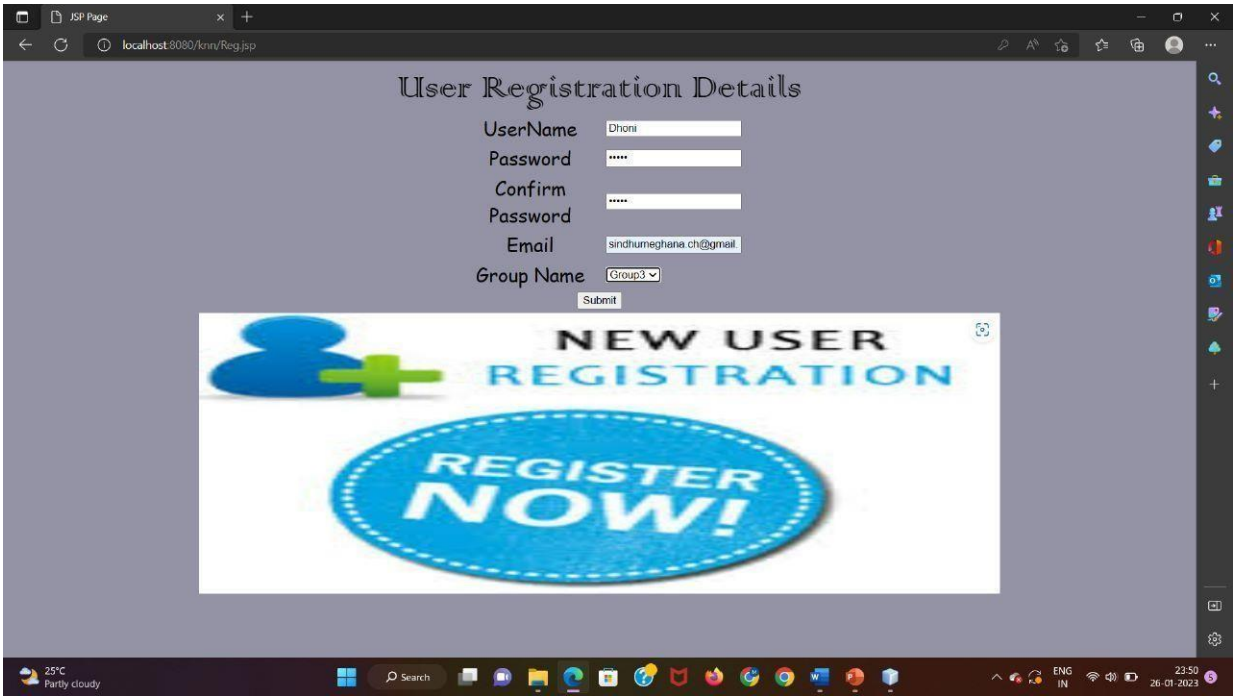
B. SCREENSHOTS:



A. Admin page



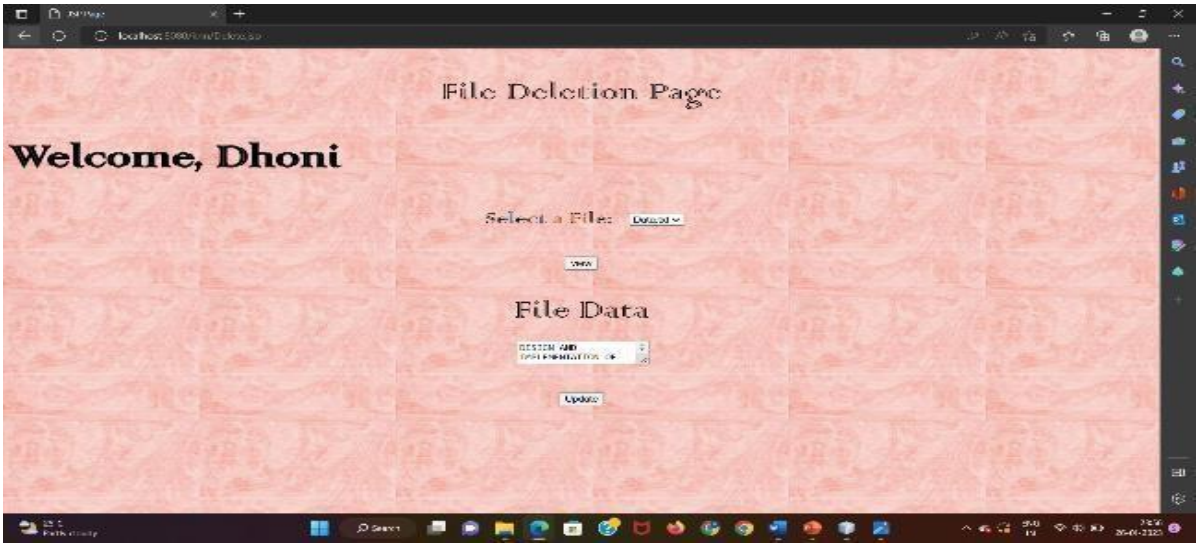
B. Login page



C. User registration page

D. Group registration page

E. File upload page



F. File deletion page

C. RESEARCH PAPER

DESIGN AND IMPLEMENTATION OF TRUST BASED ACCESS CONTROL MODEL FOR CLOUD COMPUTING

CHERUKURI SAI SINDHU MEGHANA[1]
STUDENT
DEPARTMENT OF CSE
SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
CHENNAI
sindhumeghana.ch@gmail.com

CHILUKURI SRI VARSHINI[2]
STUDENT
DEPARTMENT OF CSE
SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
CHENNAI
chillukurivarshini25@gmail.com

DR. YOVAN FELIX[3]
PROFESSOR
DEPARTMENT OF CSE
SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
CHENNAI
yovanfelix@sathyabama.ac.in

Abstract— **Shared data security is a major worry in today's world. An individual uses the data-sharing system to upload a file that has been encrypted with the private key. This feature is essential in any system for sharing huge amounts of data since it makes it hard for the owner of the data to maintain the secrecy of the information if one of the users discloses the important information. It provides a concrete and effective implementation of the strategy, showing its viability and establishing the security of the method. Data owners encounter a variety of challenges when it comes to sharing their data on servers or in the cloud. These problems have a number of solutions. These techniques are essential for handling keys that the data owner has shared. The purpose of the paper is to establish trusted authority for user authentication when accessing cloud data. A trustworthy authority generates the key using the SHA algorithm, which is subsequently sent to both the user and the owner. After receiving an AES-encrypted file from the data owner, the trusted authority module computes the hash value using the MD-5 method. Its database contains keys that will be used to detect during dynamic operations which user is abusing the system. The file is sent from a trustworthy source to the CSP module, which saves it in the cloud. It is demonstrated that the produced key sets have a variety of desired qualities that protect communication session confidentiality from collusion assaults by other network nodes. The proposed model provides a flexible and scalable approach to access control, as it allows administrators to define different trust levels and policies for different cloud resources. This model also includes mechanisms for monitoring user behavior and detecting anomalies that may indicate a breach of trust. The access control model offers an effective solution for securing cloud computing environments while ensuring that users can access the resources, they need to perform their tasks.**

Keywords— *cloud computing, resource management, cryptography, computational models, Cloud computing, Trust model, Trust value*

I. INTRODUCTION

Cloud computing is used in computer technology to describe how a computer's service is delivered. The idea behind cloud computing is that users may simply access storage, processing power, or a specially designed environment for improvement without having to worry about how they interact with it. The majority of cloud computing is internet computing. Based on how the Internet is shown in computing community diagrams, the cloud is a metaphor for the Internet; as a consequence, the cloud obscures the Internet's complexity. It is a computing technique that enables customers to receive technical services through the Internet by offering relevant sources "as a carrier." Considering the problem of having access to facts objectively, cloud computing may be seen as having both vast record structures and big cloud structures. As a result, the obtained content is of poor quality. Cloud computing has a wide variety of effects on cloud computing and huge information structures. Yet, one common factor that may be noted is the difficulty in the accurate distribution of material, a challenge that can be handled by developing measurements that attempt to improve accuracy. There are two planes in a cloud network: the control plane and the data plane. For instance, cloud computing at the records level allows computing services to be located closer to a community's edge rather than on servers in the information centre. In contrast to cloud computing, the latter places more emphasis on proximity to end users and consumer targets, dense geographic distribution and help sharing, visitor financial savings and latency reduction to enhance quality of service (QoS), as well as analytical aspect and analytical flow, which produce higher. In the case of a failure, you may utilise the results and redundancy, and you can use it in AAL situations. For offering adaptable, scalable, and economical computer services, cloud computing has grown in popularity. Yet, the development of cloud computing has also brought forth fresh security issues, especially with regard to access management. Traditional access control techniques, such as role-based access control and discretionary access control, could not sufficiently handle these concerns in cloud systems. A trust-based mechanism has been suggested as a remedy for cloud computing to get around these restrictions. It intends to provide a more fine-grained and dynamic access control system that considers user and resource trust levels in a cloud context. This model enables cloud providers to better manage and control access to sensitive data and resources, while also allowing users to access only the resources that they need. The design and implementation of a trust- based model for cloud computing involves several key components, including the establishment of trust metrics, the definition of trust policies and the development of a trust evaluation and decision-making mechanism. This model also requires the integration of various security technologies, such as authentication, authorization, and encryption. Overall, the implementation of this model for cloud computing can help to improve the security and reliability of cloud services, while also enhancing the user experience by providing more flexible and efficient access control mechanisms.

II. LITERATURE SURVEY

D.Szajda et.al. It addresses the problem of safeguarding genomic data privacy during distributed genome sequence comparison using the Smith-Waterman algorithm. This problem arises due to the sensitive nature of genomic data and the need to protect it from unauthorized access and disclosure. A data privacy approach that combines encryption and secure multi-party computing (SMC) techniques. Proposed scheme consists of several steps, including key generation, data encryption, and secure computation in the secure computation phase, the encrypted data is used to perform the Smith-Waterman algorithm using SMC techniques, such as additive sharing and multiplication protocols. To evaluate the performance

and security of their scheme using a set of experiments on a distributed network. The findings demonstrate that their plan offers a high degree of security and privacy while being feasible, efficient, and cost-effective in terms of computing and communication. It contributes to the field of data privacy in distributed genome sequence comparison by proposing a practical scheme that enables secure and private comparisons of genomic data using the Smith-Waterman algorithm. The proposed scheme can be applied in various genomic data analysis applications and can also be extended to other domains that require secure and private comparisons of sensitive data in distributed environments.[1]

X. Chen et.al. contributes to the field of secure outsourcing of modular exponentiations by proposing new algorithms that enable the computation to be performed securely and without revealing the sensitive data. The suggested methods may be used in a variety of cryptographic applications and expanded to other fields that call for the safe and private calculation of sensitive data. safe outsourcing of modular exponentiations, a key process in many cryptographic systems, is a difficulty. By shifting the calculation to more capable servers, the outsourcing of modular exponentiations might lessen the computational load on devices with limited resources, including mobile phones and smart cards. The sensitive data used in the calculation may be compromised, thus outsourcing this activity also carries a security risk. new techniques for outsourcing modular exponentiations in a safe manner. The suggested methods are based on secure multiparty computing (SMC) and homomorphic encryption, which allow the calculation to be carried out safely and without disclosing the sensitive data. The algorithms provide a high degree of security while being practical and cost-effective in terms of computing and transmission. Algorithms can also be applied in various cryptographic applications and can secure and private computation of sensitive data.[2]

According to Y. Feng et al., cloud computing is rapidly improving, and the medical community is becoming more interested in techniques for successfully releasing excessively expensive computing. Customers with limited resources may increase heavy computational workloads to a server and experience limitless computing power on a pay-as-you-go basis under the available computing paradigm. The ability to verify results is one of outsourced accounting's most crucial capabilities. Which involves comparing two sequences to determine their similarity or difference. This is a computationally intensive task that is often outsourced to cloud service providers to reduce the computational burden on clients. Outsourcing sequence comparison creates privacy and security issues, however, since the sequences being compared may include sensitive data. The authors suggest an outsourcing plan that makes use of homomorphic encryption to safeguard the confidentiality of the sequences being compared in order to allay these worries. Moreover, the scheme has a verification mechanism that enables the customer to confirm the accuracy of the outcomes provided by the cloud service provider. It offers a fascinating and ground-breaking method for outsourcing sequence comparison that takes privacy and security considerations into account while still guaranteeing accuracy and effectiveness. The authors' suggested method has the potential to be used in a variety of applications where sequence comparison is necessary, including text mining and bioinformatics.[3]

F. Kerschbaum et.al. Secure and private sequence comparisons in bioinformatics applications, where it is important to compare DNA or protein sequences to detect similarities or differences. However, the sensitive nature of these sequences makes traditional comparison methods difficult to use, as they require the sequences to be disclosed to third parties. A methodology for safe and private sequence comparisons is needed to solve this issue. The system is built on a foundation of hashing and homomorphic encryption. Key generation and sequence comparison make up the protocol's two steps. The client produces a random number and hashes it while the server generates a both key pair during key creation phase. Client uses to encrypt, use the server's public key. its sequence during sequence comparison phase and transmits it, along with the hashed value, to the server. The client may then compare the outcome with its own sequence after the server conducts homomorphic operations on the encrypted sequence and provides the result. The security and efficiency of their protocol and show that it is secure against attacks by malicious servers and efficient in terms of computation and communication costs. They also provide experimental results demonstrating the practicality of their approach for sequence comparison tasks in bioinformatics applications. It enables such comparisons without compromising the privacy of sensitive sequences. [4]

M. J. Atallah et.al. The capacity to validate is one of the key characteristics of facts outsourcing. There aren't many convenient ways to sell a series of test clients to determine if the servers are really adhering to the protocol or not. solves the issue of privacy-preserving outsourcing sequence comparisons to a third-party server. In many applications, including bioinformatics, sequence comparison is a crucial function that requires significant computer resources. The client's computing workload may be reduced by outsourcing these calculations to a third-party server; however, privacy issues are raised since the compared data may include sensitive information. Homomorphic encryption, on which the protocol is built, enables the third-party server to carry out calculations on encrypted data without first decoding it. Key generation, query generation, and query evaluation stages make up the protocol. They examine the security and effectiveness of their protocol, demonstrating that it is efficient in terms of computing and transmission costs and safe against hostile sites. Outsourcing sequence comparisons in bioinformatics applications. protocol for securely outsourcing sequence comparisons that can help to address the privacy concerns.[5]

Because of internet computing technologies like grid computing, which enable significant cooperative sharing of processing power, bandwidth, storage, and data, large-scale problems in the physical and biological sciences are changing. Once linked to such a grid, a sluggish computer equipment is no longer confined by its poor performance, limited local storage capacity, and limited bandwidth. It is instead free to take advantage of the plethora of these resources available elsewhere on the network. The fact that the data in question is frequently sensitive, such as being of national security importance, proprietary and containing commercial secrets, or must be kept private due to legal requirements such as HIPAA legislation, Gramm-Leach-Bliley, or similar laws, impedes the use of "computational outsourcing." To prevent sharing personal information, sending the results of a computation performed on such data to remote agents whose computing power is being utilised, techniques for privacy-preserving computational outsourcing are currently being developed. Safe outsourcing for generally applicable sequence comparison activities. the potential for the leak of sensitive information.

III. Proposed System

The suggested system has a relaxed chat system that might provide comfortable key distribution and verbal communication for a bustling institution. It offers a simple method for exchanging keys through nonverbal means. Without any CAs confirming the user's public key, users may safely get their non-public keys from the cluster manager. Any user inside the company can use the cloud's beginning, and users whose access has been revoked are no longer

able to access the cloud. The system can provide controlled access in detail through the user institution listing. It provides a convenient means of vocal communication that is secure against malicious attacks. Customers whose access has been revoked will no longer be able to recover their original files, even though it may be working with a malicious cloud. The layout can provide secure comments to someone using a polynomial function. The application may successfully assist dynamic organizations since it eliminates the requirement for other users' private keys to be updated and recalculated whenever a new member of a group or individual joins.

It offers safety evaluation to show the security of our scheme. Each user and resource's overall degree of trust is assessed using trust metrics as part of the trust evaluation process. Machine learning algorithms may be employed in this process to evaluate previous data and find patterns that may be utilized to forecast future behavior. The system must decide whether to give or refuse access to a user or resource based on the trust assessment. This decision may also take into account other factors such as the sensitivity of the data or resource being accessed, the user's role, and the current security posture of the cloud environment. The access control model must be integrated with the cloud infrastructure to ensure that access control policies are enforced correctly. This may involve the use of APIs and other integration mechanisms to communicate with the cloud service provider's access control mechanisms. To ensure the access control model is working effectively, the system must also provide monitoring and auditing capabilities that allow administrators to track access requests and detect any anomalous behavior. An access control model for cloud computing should be created to provide a flexible and adaptable approach to access control that takes into consideration the changing degrees of trust associated with individuals and resources as well as the dynamic nature of cloud settings. By implementing such a system, organizations can better protect their sensitive data and resources in the cloud while also improving the overall user experience.

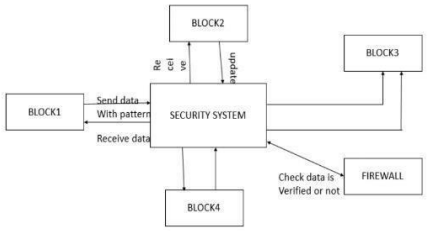


Fig. 1 Proposed System Architecture

In fig.1 it demonstrates that even when using the cloud to exchange or store data, there will be data leakage (intruders who may steal sensitive data) in order to protect this data. We use encryption and decryption algorithms, and the owner (block 1) supplies the data with a pattern that indicates encryption, and the SHA algorithm produces a key and transmits it to the security system. Block 1 delivers this key to the recipients of the data. Only by inputting that key can the Blocks 2,3,4 access the data delivered by the Block 1. After producing the key, Block 1 can send it to Blocks 2, 3, and 4. In this case, the third-party firewall determines if the data is safe. After verification, it either sends data to the security system or does not. The data will be visible to Blocks 2,3,4 who have received the encrypted key from Block 1. Without the encryption key, it becomes impossible for an intruder to collect data shared.

IV. METHODOLOGY

Identifying the issue or difficulty that the model will address is the first stage in developing and putting into practise a trust-based access control model for cloud computing. Understanding the access control specifications, such as the categories of users, resources, and activities that must be controlled, as well as any restrictions or limitations of the cloud environment, is necessary for this. This phase necessitates having a clear knowledge of the goals of the access control model, which may be accomplished by speaking with key players including cloud providers, users, and security professionals. The next stage is to create a trust-based access control model that integrates the essential trust metrics and policies to regulate access to cloud resources. This is done in accordance with the needs determined in step one and the findings of the literature research conducted in step two. Finding the metrics for measuring trust between a cloud user and a cloud resource should be the first step in the model design process. These metrics may take into account the user's standing, credentials, and prior actions as well as the resource's security status and related risk. The following step is to establish the trust policies that will be used to control access to cloud resources based on the trust metrics that have been identified; otherwise, a message indicating a failed login is displayed, and they must enter the correct information again. For new user registration, a link to registered module activity is also made available. The registered activity opens when the register button in the login activity is clicked. When registering, a new user must provide their full name, password, and phone number. The user must re-enter their password in the confirm password textbox. When a user fills out all text boxes, the data is sent to the database when they click the register button, and they are then redirected to the login process. After being uploaded to the cloud, the data owner has no control over it. In this module, the original data is encrypted into two distinct values. Before being stored in the cloud, the data in each slice can be encrypted using a variety of cryptographic algorithms and encryption keys. The Receiver can determine whether collusion is occurring or not in this Find Collusion Module by computing a distance. The receiver may also discover other parties using this Find Third-Party Module.

w

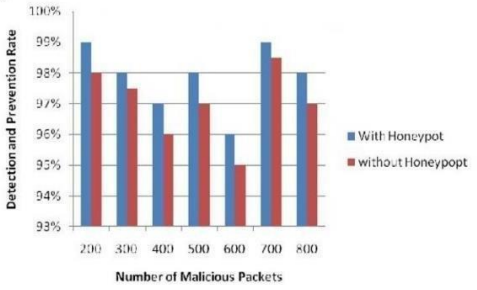


Fig.2: Attack Detection and Prevention Rate

In the fig.2: context of Control Model for Cloud Computing", this graph represents attack detection and prevention rate with honeypot and without honeypot where Y-axis represents the prevention rate and X-axis represents the attack detection in order to detect and prevent unauthorized access or malicious activity in the cloud environment. Here to evaluate the effectiveness of trust-based access control model, attack detection and prevention rates are measured. This involves monitoring the cloud environments for any unauthorized access attempts, malicious activity. Further these attack and prevention rates can then be calculated and analyzed to identify any trends or patterns in the data.

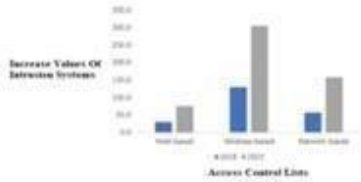


Fig.3: Intrusion Detection / Prevention System

This graph fig.3 shows the Y-axis represents the access control lists. Host system, wireless based system and network-based systems against X-axis which represents the increase in values of these intrusion systems. Here in case of host system we calculate through authentication mechanisms such as passwords or biometric. In wireless by use of secure protocol such as wifi protected access to authenticate users and encrypt data transmissions. In network-based system it is implemented through use of firewalls and prevention systems and virtual private networks. This graph shows different types of systems which provides a layered approach to security and access control can be calculated and analyzed.

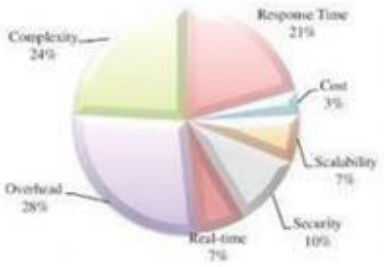


Fig.4: Intrusion detection systems in the cloud computing

This pie chart displays the percentage of various malicious activity detections to secure the system using IDS (Intrusion Detection System). Systems for detecting attacks are a crucial part of cloud computing security. In order to identify and address possible security concerns, they are used to monitor and analyse system and network activities. IDS is a cloud computing access control model that can be used to enforce access control policies by detecting unauthorised access attempts and bringing about the proper reactions, such as blocking the user or sending an alert to the system administrator. Host-based, network-based, and hybrid IDS are a few common IDS types used in cloud computing environments. Network-based IDS monitor network traffic for suspicious patterns while host-based IDS keep an eye on activity on specific machines. Host-based and network-based IDS capabilities are combined in hybrid IDS. It's crucial to take into account elements like scalability, adaptability, and compatibility with other security tools when designing an IDS for cloud computing. Cloud-based IDS solutions are becoming increasingly popular due to their ability to scale easily and handle large volumes of data. IDS plays a critical role in ensuring the security of cloud computing environments.

V.RESULTS AND DISCUSSION

Through this study, a reliable organization will be created to authenticate users who have access to cloud data. The key is created by the trustworthy authority using the SHA algorithm and then given to both the user and the owner. The data owner sends an encrypted file to the trusted authority module, which then generates a hash value using the MD-5 method. In its database, it keeps a key that will be used to locate the system's cheater during dynamic actions. A file is sent to the CSP module for cloud storage from a reliable source. Numerous advantageous characteristics of these generated key sets protect communication sessions from collusion attacks by other network nodes. One of the main benefits of an access control paradigm is enhanced security for cloud services. The model enables more precise control over who has access to what information and resources, which can help to avoid unauthorized access and data breaches. Security can be further improved by integrating trust metrics and evaluation processes to make sure that only reliable users and resources are given access. The access control model aims to give cloud services more adaptable access control mechanisms. Users may be able to access only the resources they require as a result, and cloud providers will be able to manage and control access more effectively. The use of machine learning algorithms in the trust evaluation process can help to make access control decisions more adaptive and responsive to changing conditions. This model can help to reduce the management complexity of access control in cloud environments. By automating trust evaluation and decision-making processes, administrators can focus on higher-level security tasks such as policy definition and monitoring. The model also allows for centralized management of access control policies across multiple cloud services. In summary, implementing a trust-based access control paradigm for cloud computing may have a number of advantages, such as better security, more adaptability, and less administration complexity. To make sure the model is operating successfully and efficiently, however, much thought must go into its design and execution as well as regular review and monitoring.

FIG 5: Registration

The fig 5 shows the admin login page where the admin has to provide the user's name and password then registration process where the admin has to create the group name and group key. With the help of this group key user can have access to the data shared by the admin. The user registration where the user has to fill the user details along with the group name that the user assigned to.

FIG 6: File Download Page

The fig 6 shows the download page where the user has to select the file which user wants to share the data after selecting the required file the user has to provide the eleven-digit key which the user created in group registration process. Now this data can be seen only

to the users who have access to the public key provided by the admin.

FIG 7: Update Page

The fig 7 shows the file deletion page here and can view the uploaded document. Only the users who have the public key provided by the admin can delete or update and make any changes to the file or data. The intruders can't peek into the data means, can't have access to the data which means intruder can't view or update the file without the key.

VI. Conclusion

According to the above summary, a trusted authority to verify a user who has access to cloud data is necessary owing to the pervasive issue of collusion assaults in the secure outsourcing of sequence comparison algorithms. The key is created by the trustworthy authority using the SHA algorithm and is then sent to both the user and the owner. The trusted authority module computes the hash value using the MD-5 method after receiving an AES-encrypted file from the data owner. Its database includes keys that may be used to distinguish the system's third party during dynamic operations. The answer to access control in cloud computing settings is a trust-based access control approach. It provides more fine-grained and dynamic access control mechanisms that take into account the trust level of users and resources. By integrating trust metrics, trust policies, and trust evaluation processes, access control can improve security, increase flexibility, and reduce management complexity. However, the implementation of a trust-based access control model must be carefully designed and evaluated to ensure its effectiveness and efficiency. As well as concerns around the privacy and security of data, must be carefully considered. Ongoing monitoring are also essential to ensure that the access control model is working as intended and to identify any potential issues or areas for improvement. Overall, a trust-based access control model can help organizations to better manage and control access to their cloud resources while also enhancing the user experience by providing more flexible and efficient access control mechanisms. As cloud computing continues to evolve and become more prevalent, the development and implementation of effective access models will be critical to ensuring the security and reliability of cloud services.

VII. References

[1] D. Szajda, M. Pohl, J. Owen, and B. Lawson, "Toward a practical data privacy scheme for a distributed implementation of the Smith-Waterman genome sequence comparison algorithm," in Proc. Netw. Distrib. Syst. Secur. Symp. (NDSS), San Diego, CA, USA, 2006, pp. 253–265.

[2] X. Chen, J. Li, J. Ma, Q. Tang, and W. Lou, "New algorithms for secure outsourcing of modular exponentiations," IEEE Trans. Parallel Distrib. Syst., vol. 25, no. 9, pp. 2386–2396, Sep. 2014.

[3] Y. Feng, H. Ma, and X. Chen, 'Efficient and verifiable outsourcing scheme of sequence comparisons," Intell. Autom. Soft Comput., vol. 21, no. 1, pp. 51–63, Jan. 2015.

[4] M. J. Atallah, F. Kerschbaum, and W. Du, "Secure and private sequence comparisons," in Proc. ACM Workshop Privacy Electron. Soc. (WPES), Washington, DC, USA, 2003, pp. 39–44.

[5] M. J. Atallah and J. Li, "Secure outsourcing of sequence comparisons," in Proc. Int. Workshop Privacy Enhancing Technol. (PET), Toronto, ON, Canada, 2004, pp. 63–78.

[6] A. Alotaibi, M. Khan, and M. Alghamdi, "Trust-Based Access Control Model for Cloud Computing," International Journal of Advanced Computer Science and Applications, vol. 10, no. 4, pp. 356-364, 2019.

[7] N. E. Fenton, "Trust-Based Access Control in the Cloud," Journal of Cloud Computing: Advances, Systems and Applications, vol. 1, no. 1, 2012.

[8] S. A. Kumar and K. S. Ramaswamy, "Design and Implementation of a Trust-Based Access Control Model for Cloud Computing," Journal of Information Security, vol. 7, no. 3, pp. 103-112, 2016.

[9] Qian, Y. Zhu, Y. Yang, and L. Wang, "A Trust-Based Access Control Model for Cloud Computing," in Proceedings of the 2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications, pp. 90-97, 2012.

[10] Singh, R. Gupta, and M. Kumar, "Trust-Based Access Control in Cloud Computing: A Review," in Proceedings of the 2014 International Conference on Advances in Computing, Communications and Informatics, pp. 1188-1195, 2014.

[11] K. Liu, Y. Liu, and J. Zhang, "A Trust-Based Access Control Model for Cloud Computing," in Proceedings of the 2015 International Conference on Cloud Computing and Big Data, pp. 101-107, 2015.