**ABSTRACT**

The demand for remote data storage and computation services is increasing exponentially in our data-driven society; thus, the need for secure access to such data and services. In this paper, we design a new biometric-based authentication protocol to provide secure access to a remote (cloud) server. In the proposed approach, we consider biometric data of a user as a secret credential. We then derive a unique identity from the user’s biometric data, which is further used to generate the user’s private key. In addition, we propose an efficient approach to generate a session key between two communicating parties using two biometric templates for a secure message transmission. In other words, there is no need to store the user’s private key anywhere and the session key is generated without sharing any prior information. A detailed Real-Or- Random (ROR) model based formal security analysis, informal (non-mathematical) security analysis and also formal security verification using the broadly-accepted Automated Validation of Internet Security Protocols and Applications (AVISPA) tool reveal that the proposed approach can resist several known attacks against (passive/active) adversary. Finally, extensive experiments and a comparative study demonstrate the efficiency and utility of the proposed approach.

Index Terms—Authentication, biometric-based security, cloud service access, session key.

**1.INTRODUCTION**

Cloud services are a norm in our society. However, providing secure access to cloud services is not a trivial task, and designing robust authentication, authorization and accounting for access is an ongoing challenge, both operationally and research-wise. A number of authentication mechanisms have been proposed in the literature, such as those based on Kerberos [1], OAuth [2] and OpenID [3] (see [1], [4]– [12]). Generally, these protocols seek to establish a secure delegated access mechanism among two communicating entities connected in a distributed system. These protocols are based on the underlying assumption that the remote server responsible for authentication is a trusted entity in the network. Specifically, a user first registers with a remote server. This is needed to ensure the authorization of the owner. When a user wishes to access a server, the remote server authenticates the user and the user also authenticates the server. Once both verifications are successfully carried out, the user obtains access to the services from some remote server. One key limitation in existing authentication mechanisms is that the user’s credentials are stored in the authentication server, which can be stolen and (mis)used to gain unauthorized access to various services. Also, to ensure secure and fast communication, existing mechanisms generally use symmetric key cryptography, which requires a number of cryptographic keys to be shared during the authentication process. This strategy results in an overhead to the authentication protocols. Designing secure and efficient authentication protocols is challenging, as evidenced by the weaknesses revealed in the published protocols of Jiang et al. [13], Althobaiti et al. [14], Xue et al. [15], Turkanovic et al. [16], Park et al. [17], Dhillon and Kalra [18], Kaul and Awasthi [19] and Kang et al. [20] – see also Section II. Therefore, in this paper we seek to design a secure and efficient authentication protocol. Specifically, we will first provide an alternative to conventional password-based authentication mechanism. Then, we demonstrate how one can build a secure communication between communicating parties involved in the authentication protocol, without having any secret pre-loaded (i.e., shared) information. In the proposed approach, we consider a fingerprint image of a user as a secret credential. From the fingerprint image, we generate a private key that is used to enroll the user’s credential secretly in the database of an authentication server. In the authentication phase, we capture a new biometric fingerprint image of the user, and subsequently generate the private key and encrypt the biometric data as a query. This queried biometric data is then transmitted to the authentication server for matching with the stored data. Once the user is authenticated successfully, he/she is ready to access his/her service from the desired server. To obtain secure access to the service server, mutual authentication between the user and authentication server, and also between the user and service server have been proposed using a short-term session key. Using two fingerprint data, we present a fast and robust approach to generate the session key. In addition, a biometricbased message authenticator is also generated for message authenticity purpose.We summarize the key contributions/benefits related to the proposed approach as below. 1) An effective way to transmit the user’s biometric data through the unsecured network channels to an authentication server is presented. 2) We propose an approach to generate a revocable private key directly from an irrevocable fingerprint image. There is no need to store the private key or a direct form of the user’s biometric data anywhere. 3) We mitigate the limitation in traditional mechanisms that require the user’s credentials to be stored in the authentication server. 4) We introduce a novel way to generate session keys. 5) In traditional authentication protocol, each entity requires some preloaded information; thus, incurring some overhead. We introduce a new mechanism to avoid the need for secret pre-loaded information. 6) A message authentication mechanism, as an alternative to the existing message authentication protocols (i.e., Message Authentication Code (MAC)), is introduced.

**2.LITERATURE SURVEY**

**2.1 Biometrics-Based Privacy-Preserving User Authentication Scheme for Cloud-Based Industrial Internet of Things Deployment**

**AUTHORS: A. K. Das, M. Wazid, N. Kumar, A. V. Vasilakos, and J. J. P. C. Rodrigues,**

**ABSTRACT:**

Due to the widespread popularity of Internet-enabled devices, Industrial Internet of Things (IIoT) becomes popular in recent years. However, as the smart devices share the information with each other using an open channel, i.e., Internet, so security and privacy of the shared information remains a paramount concern. There exist some solutions in the literature for preserving security and privacy in IIoT environment. However, due to their heavy computation and communication overheads, these solutions may not be applicable to wide category of applications in IIoT environment. Hence, in this paper, we propose a new biometric-based privacy preserving user authentication (BP2UA) scheme for cloud-based IIoT deployment. BP2UA consists of strong authentication between users and smart devices using preestablished key agreement between smart devices and the gateway node. The formal security analysis of BP2UA using the well-known real-or-random model is provided to prove its session key security. Moreover, an informal security analysis of BP2UA is also given to show its robustness against various types of known attacks. The computation and communication costs of BP2UA in comparison to the other existing schemes of its category demonstrate its effectiveness in the IIoT environment. Finally, the practical demonstration of BP2UA is also done using the NS2 simulation.

**2.2** **Security and Accuracy of Fingerprint-Based Biometrics: A Review**

**AUTHORS:** **W. Yang, S. Wang, J. Hu, G. Zheng, and C. Valli**

**ABSTRACT:** Biometric systems are increasingly replacing traditional password- and token-based authentication systems. Security and recognition accuracy are the two most important aspects to consider in designing a biometric system. In this paper, a comprehensive review is presented to shed light on the latest developments in the study of fingerprint-based biometrics covering these two aspects with a view to improving system security and recognition accuracy. Based on a thorough analysis and discussion, limitations of existing research work are outlined and suggestions for future work are provided. It is shown in the paper that researchers continue to face challenges in tackling the two most critical attacks to biometric systems, namely, attacks to the user interface and template databases. How to design proper countermeasures to thwart these attacks, thereby providing strong security and yet at the same time maintaining high recognition accuracy, is a hot research topic currently, as well as in the foreseeable future. Moreover, recognition accuracy under non-ideal conditions is more likely to be unsatisfactory and thus needs particular attention in biometric system design. Related challenges and current research trends are also outlined in this paper.

**2.3 Difference co-occurrence matrix using BP neural network for fingerprint liveness detection**

# AUTHORS: C. Yuan, X. Sun, and Q. M. J. Wu

**ABSTRACT:** With the growing use of fingerprint identification systems in recent years, preventing fingerprint identification systems from being spoofed by artificial fake fingerprints has become a critical problem. In this paper, we put forward a novel method to detect fingerprint liveness based on BP neural network, which is used for the first time in the fingerprint liveness detection. Moreover, different from traditional detection methods, we propose a scheme to construct the input data and corresponding category labels. More effective and efficient texture features of fingerprints, which are used as the input data of the BP neural network, are computed to improve classification performance and obtain a better pre-trained network model. After a variety of preprocessing operations and image compression operations, gradient values in the horizontal and vertical directions are computed by using Laplacian operator, and difference co-occurrence matrices are constructed from the obtained gradient values. Then, the input data of neural network model are built based on two DCMs. The pre-trained neural network models with diverse neuron nodes are learnt. Different experiments based on different parameters for the BP neural network have been conducted. Finally, classification accuracy of testing fingerprints is predicted based on the pre-trained networks. Experimental results on the LivDet 2013 show that the classification performance of our proposed method is effective and meanwhile provides a better detection accuracy compared with the majority of previously published results.

# 2.4 An Untraceable Biometric-Based Multi-server Authenticated Key Agreement Protocol with Revocation

# AUTHORS: C.-C. Chang and N.-T. Nguyen,

**ABSTRACT:** Online access has been widely adopted to distribute diversified services to customers. In this architecture, public channels are utilized to exchange information between end users and remote servers at anytime and anywhere. To achieve confidentiality and integrity for transferred data, the related parties have to authenticate each other and negotiate a secret session key to encrypt and decrypt exchanged messages. Since the Lamport’s pioneering authentication work in 1981, numerous mechanisms have been proposed to enhance security as well as reduce computation and payload data. Recently, Chuang and Chen proposed a multi-server authenticated agreement protocol employing a smart card and biometric data to eliminate the weaknesses caused by parameters related to low-entropy human-memorable passwords that are stored in a physical location. However, Mishra et al. showed that Chuang and Chen’s protocol is not only vulnerable to multiple attacks but also suffers from the drawback of variation of biometric data. To overcome these weaknesses, they proposed an enhanced three-factor authenticated key agreement protocol using the low-error rate Biohashing technique. Unfortunately, we found that Mishra et al.’s scheme is also vulnerable to the denial-of-service attack, the traceable user attack, the impersonation attack, and the pre-shared key attack. Furthermore, the protocol does not provide any user revocation mechanism to control user accesses. In this novel untraceable authenticated key agreement scheme, we adopt the Hamming distance to verify encrypted Biohash codes and a public-key technique to construct the revocation mechanism. Our scheme achieves not only zero errors of biometric verification but also secure against all known attacks.

**2.5**  **A secure temporal-credentialbased mutual authentication and key agreement scheme for wireless sensor networks**

**AUTHORS:** **D. He, N. Kumar, and N. Chilamkurti,**

**ABSTRACT:** With the development of wireless communication technology and sensor technology, the wireless sensor network (WSN) has been widely used in various applications, such as military surveillance, environment monitoring industry control, medical monitoring, and so on. In most of the cases, WSNs are deployed in unattended environment. So, these are more vulnerable to various attacks than traditional networks. To protect communications in WSNs, mutual authentication and key agreement schemes for WSNs have been studied widely. Recently, Xue et al. proposed a temporal-credential-based mutual authentication and key agreement scheme for WSNs and claimed their scheme could withstand various attacks. However, in this paper, we will point out that their scheme is vulnerable to the off-line password guessing attack, the user impersonation attack, the sensor node impersonation attack and the modification attack. To overcome weaknesses in Xue et al.'s scheme, we also propose a new temporal-credential-based mutual authentication and key agreement scheme for WSNs. Security analysis shows our scheme could overcome weaknesses in Xue et al.'s scheme. Performance analysis shows our scheme also has better performance. Therefore, our scheme is more suitable for providing secure communication in WSNs.

**3.SYSTEM ANALYSIS**

**3.1EXISTING SYSTEM:**

A number of authentication mechanisms have been proposed in the literature, such as those based on Kerberos [1], OAuth [2] and OpenID [3] (see [1], [4]– [12]). Generally, these protocols seek to establish a secure delegated access mechanism among two communicating entities connected in a distributed system. These protocols are based on the underlying assumption that the remote server responsible for authentication is a trusted entity in the network. Specifically, a user first registers with a remote server. This is needed to ensure the authorization of the owner. When a user wishes to access a server, the remote server authenticates the user and the user also authenticates the server. Once both verifications are successfully carried out, the user obtains access to the services from some remote server.

One key limitation in existing authentication mechanisms is that the user’s credentials are stored in the authentication server, which can be stolen and (mis)used to gain unauthorized access to various services. Also, to ensure secure and fast communication, existing mechanisms generally use symmetric key cryptography, which requires a number of cryptographic keys to be shared during the authentication process. This strategy results in an overhead to the authentication protocols. Designing secure and efficient authentication protocols is challenging, as evidenced by the weaknesses revealed in the published protocols of Jiang et al. [13], Althobaiti et al. [14], Xue et al. [15], Turkanovic et al. [16], Park et al. [17], Dhillon and Kalra [18], Kaul and Awasthi [19] and Kang et al. [20] – see also Section II. Therefore, in this paper we seek to design a secure and efficient authentication protocol. Specifically, we will first provide an alternative to conventional password-based authentication mechanism. Then, we demonstrate how one can build a secure communication between communicating parties involved in the authentication protocol, without having any secret pre-loaded (i.e., shared) information.

Disadvantages:

1.In existing authentication mechanisms is that the user’s credentials are stored in the authentication server, which can be stolen and (mis)used to gain unauthorized access to various services.

2. When a user wishes to access a server, the remote server authenticates the user and the user also authenticates the server.

* 1. **PROPOSED SYSTEM:**

In the proposed approach, we consider a fingerprint image of a user as a secret credential. From the fingerprint image, we generate a private key that is used to enroll the user’s credential secretly in the database of an authentication server. In the authentication phase, we capture a new biometric fingerprint image of the user, and subsequently generate the private key and encrypt the biometric data as a query. This queried biometric data is then transmitted to the authentication server for matching with the stored data. Once the user is authenticated successfully, he/she is ready to access his/her service from the desired server. To obtain secure access to the service server, mutual authentication between the user and authentication server, and also between the user and service server have been proposed using a short-term session key. Using two fingerprint data, we present a fast and robust approach to generate the session key. In addition, a biometricbased message authenticator is also generated for message authenticity purpose.

We summarize the key contributions/benefits related to the proposed approach as below.

1) An effective way to transmit the user’s biometric data through the unsecured network channels to an authentication server is presented.

2) We propose an approach to generate a revocable private key directly from an irrevocable fingerprint image. There is no need to store the private key or a direct form of the user’s biometric data anywhere.

3) We mitigate the limitation in traditional mechanisms that require the user’s credentials to be stored in the authentication server.

4) We introduce a novel way to generate session keys.

5) In traditional authentication protocol, each entity requires some preloaded information; thus, incurring some overhead. We introduce a new mechanism to avoid the need for secret pre-loaded information.

6) A message authentication mechanism, as an alternative to the existing message authentication protocols (i.e., Message Authentication Code (MAC)), is introduced.

**Advantages:**

1. An efficient approach to generate a session key between two communicating parties using two biometric templates for a secure message transmission.

2.There is no need to store the user’s private key anywhere and the session key is generated without sharing any prior information.

* 1. **SYSTEM REQUIREMENTS:**

This section elaborates on the functional requirements of the application. The SRS itself can be divided into module, each module having specifications. In order to carry out the project, the following hardware and software is required.

**HARDWARE REQUIREMENTS:**

* System : i3
* Hard Disk : 40 GB.
* Floppy Drive : 1.44 Mb.
* Monitor : 15 VGA Colour.
* Mouse : Logitech.
* Ram : 512 Mb.

**SOFTWARE REQUIREMENTS:**

**Software Requirements**

**Technology** **:**  Java 2 Standard Edition, JDBC

**Web** **Server**  **:** Tomcat 7.0

**Client Side Technologies :** HTML, CSS, JavaScript

**Server Side Technologies :** Servlets, JSP

**Data Base Server :** MySQL

**Editor :** Netbeans8.1

**3.4 SYSTEM STUDY**

**FEASIBILITY STUDY**

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.

Three key considerations involved in the feasibility analysis are

* ECONOMICAL FEASIBILITY
* TECHNICAL FEASIBILITY
* SOCIAL FEASIBILITY

**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 be justified. 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. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

**4.SYSTEM DESIGN**

**4.1 SYSTEM ARCHITECTURE:**

****

**4.2 UML DIAGRAMS**

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

**GOALS:**

The Primary goals in the design of the UML are as follows:

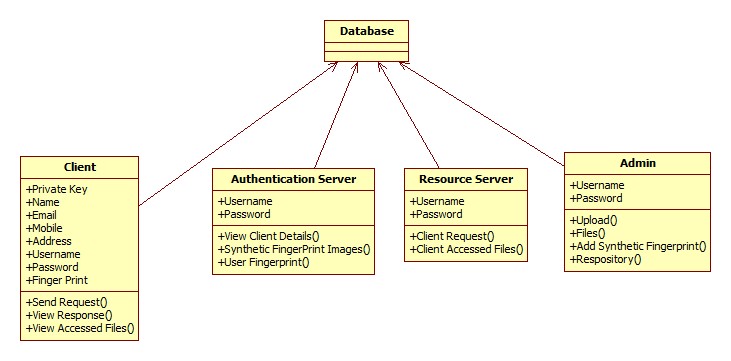
1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

**USE CASE DIAGRAM:**

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

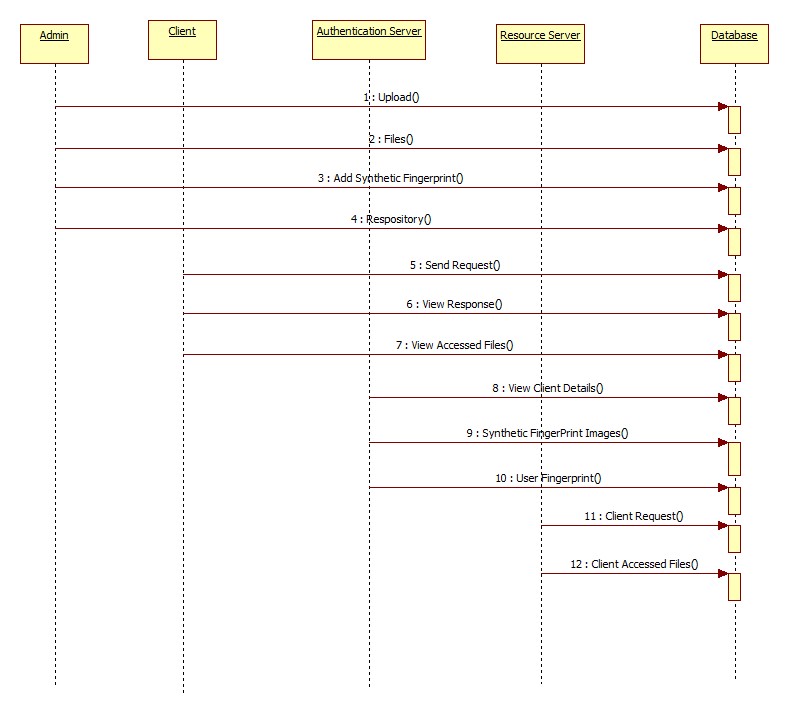
**CLASS DIAGRAM:**

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



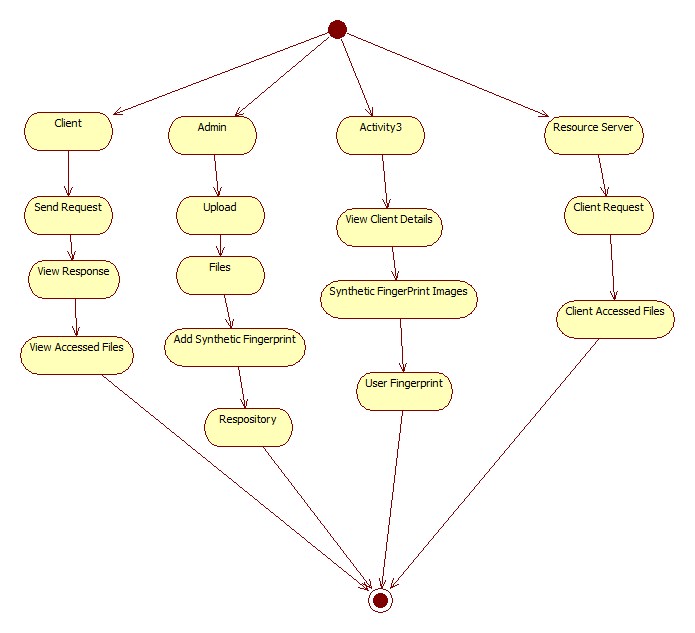
**SEQUENCE DIAGRAM:**

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

****

**ACTIVITY DIAGRAM:**

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



**5.SYSTEM DESIGN**

System design is transition from a user oriented document to programmers or data base personnel. The design is a solution, how to approach to the creation of a new system. This is composed of several steps. It provides the understanding and procedural details necessary for implementing the system recommended in the feasibility study. Designing goes through logical and physical stages of development, logical design reviews the present physical system, prepare input and output specification, details of implementation plan and prepare a logical design walkthrough.

The database tables are designed by analyzing functions involved in the system and format of the fields is also designed. The fields in the database tables should define their role in the system. The unnecessary fields should be avoided because it affects the storage areas of the system. Then in the input and output screen design, the design should be made user friendly. The menu should be precise and compact.

**SOFTWARE DESIGN**

In designing the software following principles are followed:

1. **Modularity and partitioning**: software is designed such that, each system should consists of hierarchy of modules and serve to partition into separate function.

2. **Coupling:** modules should have little dependence on other modules of a system.

3. **Cohesion:** modules should carry out in a single processing function.

4. **Shared use:** avoid duplication by allowing a single module be called by other that need the function it provides

**MODULE DESIGN:**

**1.CLIENT**

**2.AUTHENTICATION SERVER**

**3.ADMIN**

**4.RESOURCE SERVER**

**1.CLIENT**

**Client has to register into application with basic details and he can able to login with username ,password and with fingerprint. Client can able sent request to the resource server. After sending the request he can get the response from the resource server.after getting the response from the server he can able view the file in the cloud.He can able to see all permission of files.**

**2. AUTHENTICATION SERVER.**

**Authentication Server need to login with username and password. After login he can able to view client details and authorize . Authentication server can able to view synthetic finger print images. Server can able to user client images.**

**3.ADMIN**

**Admin need to login with basic username and password. After login he can able to upload files those are useful to the user. He can able to view all uploaded files. Admin can able to add synthetic fingerprint images.Admin can able to view the data in the repository.**

**4.RESOURCE SERVER**

**Resource server need to login into the application using username and password. After login resource server he can able to view all client requests as well as he can able view all users access rights of files.**

**INPUT/OUTPUT DESIGN**

**Input design:** considering the requirements, procedures to collect the necessary input data in most efficiently designed. The input design has been done keeping in view that, the interaction of the user with the system being the most effective and simplified way.

Also the measures are taken for the following

* + - Controlling the amount of input
    - Avoid unauthorized access to the classroom.
    - Eliminating extra steps
    - Keeping the process simple
    - At this stage the input forms and screens are designed.

**Output design:** All the screens of the system are designed with a view to provide the user with easy operations in simpler and efficient way, minimum key strokes possible. Instructions and important information is emphasized on the screen. Almost every screen is provided with no error and important messages and option selection facilitates. Emphasis is given for speedy processing and speedy transaction between the screens. Each screen assigned to make it as much user friendly as possible by using interactive procedures. So to say user can operate the system without much help from the operating manual.

**6.OVERVIEW OF SOFTWARE DEVELOPMENT TOOLS**

**HTML**

Html is a language which is used to create web pages with html marking up a page to indicate its format, telling the web browser where you want a new line to begin or how you want text or images aligned and more are possible.

We used the following tags in our project.

**Table:**

Tables are so popular with web page authors is that they let you arrange the elements of a web page in such a way that the browser won’t rearrange them web page authors frequently use tables to structure web pages.

**TR:**

TRis used to create a row in a table encloses <TH> and <TD> elements. <TR> contain many attributes. Some of them are,

* ALIGN: specifies the horizontal alignment of the text in the table row.
* BGCOLOR: Specifies the background color for the row.
* BORDERCOLOR: Sets the external border color for the row.
* VALIGN: Sets the vertical alignment of the data in this row.

**TH:**

TH is used to create table heading.

* ALIGN: Sets the horizontal alignment of the content in the table cell. Sets LEFT, RIGHT, CENTER.
* BACKGROUND: Species the back ground image for the table cell.
* BGCOLOR: Specifies the background color of the table cell
* VALIGN: Sets the vertical alignment of the data. Sets to TOP, MIDDLE, BOTTOM or BASELINE.
* WIDTH: Specifies the width of the cell. Set to a pixel width or a percentage of the display area.

**TD:**

TD is used to create table data that appears in the cells of a table.

* ALIGN: Species the horizontal alignment of content in the table cell. Sets to LEFT, CENTER, RIGHT.
* BGCOLOR: Specifies the background image for the table cell.
* BGCOLOR: sets the background color of the table cells.
* WIDTH: Species the width of the cell

**Frames:**

Frames are used for either run off the page or display only small slices of what are supposed to be shown and to configure the frame we can use <FRAMESET>There are two important points to consider when working with <FRAMESET>.

* <FRAMESET> element actually takes the place of the <BODY> element in a document.
* Specifying actual pixel dimensions for frames .

<FRAME> Elements are used to create actual frames.

From the frameset point of view dividing the browser into tow vertical frames means creating two columns using the <FRAMESET> elements COLS attribute.

The syntax for vertical fragmentation is,

<FRAMESET COLS =”50%, 50%”>

</FRAMESET>

Similarly if we replace COLS with ROWS then we get horizontal fragmentation.

The syntax for horizontal fragmentation is,

<FRAMESET ROWS=”50%, 50%”>

</FRAMESET>

**Form:**

The purpose of FORM is to create an HTML form; used to enclose HTML controls, like buttons and text fields.

**Attribute:**

* ACTION: Gives the URL that will handle the form data.
* NAME: Gives the name to the form so you can reference it in code set to an alphanumeric string.
* METHOD: method or protocol is used to sending data to the target action URL. The GET method is the default, it is used to send all form name/value pair information in an URL. Using the POST method, the content of the form are encoded as with the GET method, but are sent in environment variables.

## Controls in HTML:

**<**INPUT TYPE =BUTTON>:

Creates an html button in a form.

ATTRIBUTES:

* NAME: gives the element a name. Set to alphanumeric characters.
* SIZE: sets the size.
* VALUE: sets the caption of the element.

**<**INPUT TYPE = PASSWORD>:

Creates a password text field, which makes typed input.

ATTRIBUTES:

* NAME: gives the element a name, set to alphanumeric characters.
* VALUE: sets the default content of the element.

<INPUT TYPE=RADIO>:

**C**reates a radio button in a form.

ATTRIBUTE:

* NAME: Gives the element a name. Set to alphanumeric character.
* VALUE: Sets the default content of the element.

<INPUT TYPE=SUBMIT>:

Creates a submit button that the user can click to send data in the form back to the web server.

ATTRIBUTES:

NAME: Gives the element a name. Set to alphanumeric characters.

VALUE: Gives this button another label besides the default, Submit Query. Set to alphanumeric characters.

**<**INPUT TYPE=TEXT>:

Creates a text field that the user can enter or edit text in.

ATTRIBUTES:

NAME: Gives the element a name. Set to alphanumeric characters.

VALUE: Holds the initial text in the text field. Set to alphanumeric characters.

**Java Script:**

Java script originally supported by Netscape navigator is the most popular web scripting language today. Java script lets you embedded programs right in your web pages and run these programs using the web browser. You place these programs in a <SCRIPT> element, usually within the <HEAD> element. If you want the script to write directly to the web page, place it in the <BODY> element.

**Java script Methods:**

**Writeln:**

Document.writeln () is a method, which is used to write some text to the current web page.

**onClick:**

Occurs when an element is clicked.

**onLoad:**

Occurs when the page loads.

**onMouseDown:**

Occurs when a mouse button goes down.

**onMouseMove:**

Occurs when the mouse moves.

**OnUnload:**

Occurs when a page is unloaded.

**MySQL:**

MySQL is an open source relational database management system (RDBMS).This is the most popular database system used with PHP. MySQL is distributed and supported by Oracle Corporation.

MySQL runs on almost all platforms including Linux, Unix and Windows. Although it can be used in a wide range of applications, MySQL is often associated with web applications and online publishing.

MySQL is an essential constituent of an open source enterprise stack called LAMP. LAMP is a web development platform that uses Linux as an operating system, in the form of Apache web server, MySQL relational database management system and PHP object-oriented scripting language.

**Advantages of MySQL:**

**Data Security:** MySQL is globally renowned for being the most secure and reliable database management system used in popular web applications including WordPress, Drupal, Joomla, Facebook and Twitter.

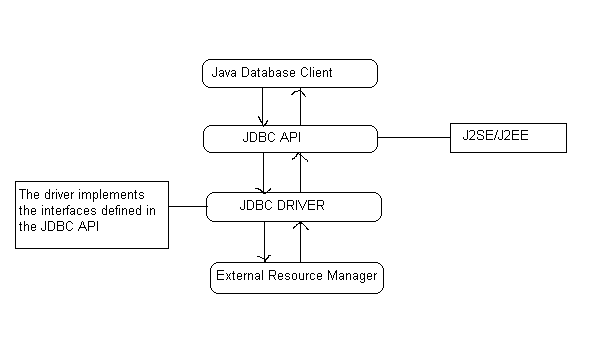
**High Performance**: MySQL features a distinct storage-engine framework that facilitates system administrators to configure the MySQL database server for a flawless performance.

**Round-the-Clock Up-time:** MySQL comes with the assurance of 24×7 up-time and offers a wide range of high-availability solutions, including specialized cluster servers and master/slave replication configurations.

**The Flexibility of Open Source:** All the fears and worries that arise in an open-source solution can be brought to an end with MySQL’s round-the-clock support and enterprise indemnification. The secure processing and trusted software of MySQL combine to provide effective transactions for large-volume projects. It makes maintenance, debugging and upgrades fast and easy while enhancing the end-user experience.

**JDBC Drivers:**

The JDBC API only defines interfaces for objects used for performing various database-related tasks like opening and closing connections, executing SQL commands, and retrieving the results. We all write our programs to interfaces and not implementations. Either the resource manager vendor or a third party provides the implementation classes for the standard JDBC interfaces. These software implementations are called JDBC drivers.JDBC drivers transform the standard JDBC calls to the external resource manager-specific API calls. The diagram below depicts how a database client written in java accesses an external resource manager using the JDBC API

**driver:**

Depending on the mechanism of implementation, JDBC drivers are broadly classified into four types.

**TYPE1:**

Type1 JDBC drivers implement the JDBC API on top of a lower level API like ODBC. These drivers are not generally portable because of the independency on native libraries. These drivers translate the JDBC calls to ODBC calls and ODBC sends the request to external data source using native library calls. The JDBC-ODBC driver that comes with the software distribution for J2SE is an example of a type1 driver.

**TYPE2:**

Type2 drivers are written in mixture of java and native code. Type2 drivers use vendors specific native APIs for accessing the data source. These drivers transform the JDBC calls to vendor specific calls using the vendor’s native library.

These drivers are also not portable like type1 drivers because of the dependency on native code.

**TYPE3:**

Type3 drivers use an intermediate middleware server for accessing the external data sources. The calls to the middleware server are database independent. However, the middleware server makes vendor specific native calls for accessing the data source. In this case, the driver is purely written in java.

**TYPE4:**

Type4 drivers are written in pure java and implement the JDBC interfaces and translate the JDBC specific calls to vendor specific access calls. They implement the data transfer and network protocol for the target resource manager. Most of the leading database vendors provide type4 drivers for accessing their database servers.

**Driver manager and Driver:**

The java.sql package defines an interface called Java.sql.Driver that makes to be implemented by all the JDBC drivers and a class called java.sql.DriverManager that acts as the interface to the database clients for performing tasks like connecting to external resource managers, and setting log streams. When a JDBC client requests the DriverManager to make a connection to an external resource manager, it delegates the task to an approate driver class implemented by the JDBC driver provided either by the resource manager vendor or a third party.

**JAVA.SQL.DRIVERMANAGER:**

The primary task of the class driver manager is to manage the various JDBC drivers register. It also provides methods for:

* Getting connections to the databases.
* Managing JDBC logs.
* Setting login timeout.

**Managing drivers:**

JDBC clients specify the JDBC URL when they request a connection. The driver manager can find a driver that matches the request URL from the list of register drivers and delegate the connection request to that driver if it finds a match JDBC URLs normally take the following format:

**<protocol>:<sub-protocol>:<resource>**

The protocol is always jdbc and the sub-protocol and resource depend on the type of resource manager. The URL for postgreSQL is in the format:

**Jdbc: postgres ://< host> :< port>/<database>**

Here host is the host address on which post master is running and database is the name of the database to which the client wishes to connect.

**Managing controls:**

DriverManager class is responsible for managing connections to the databases:

public static Connection getConnection (String url,Properties info) throws SQLException

This method gets a connection to the database by the specified JDBC URL using the specified username and password. This method throws an instance of SQLException if a database access error occurs.

**Connetions:**

The interface java.sql.Connection defines the methods required for a persistent connection to the database. The JDBC driver vendor implements this interface. A database ‘vendor-neutral’ client never uses the implementation class and will always use only the interface. This interface defines methods for the following tasks:

* Statements, prepared statements, and callable statements are the different types of statements for issuing sql statements to the database by the JDBC clients.
* For getting and setting auto-commit mode.
* Getting meta information about the database.
* Committing and rolling back transactions.

**Creating connections:**

The interface java.sql.Connection defines a set of methods for creating database statements. Database statements are used for sending SQL statements to the database:

Public Statement createStatement () throws SQLException

This method is used for creating instances of the interface java.sql.Statement. This interface can be used for sending SQL statements to the database. The interface java.sql.Statement is normally used for sending SQL statements that don’t take any arguments. This method throws an instance of SQLException if a database access error occurs:

Public Statement createStatement (int resType, int resConcurrency) throws SQLException.

**JDBC resultset:**

A JDBC resultset represents a two dimentional array of data produced as a result of executing SQL SELECT statements against databases using JDBC statements. JDBC resultsets are represented by the interface java.sql.ResultSet. The JDBC vendor provider provides the implementation class for this interface.

**Scrolling resultset:**

public boolean next() throws SQLException

public boolean previous() throws SQLException

public boolean first() throws SQLException

public boolean last() throws SQLException

**Statement:**

The interface java.sql.stament is normally used for sending SQL statements that do not have IN or OUT parameters. The JDBC driver vendor provides the implementation class for this interface. The common methods required by the different JDBC statements are defined in this interface. The methods defined by java.sql. Statement can be broadly categorized as follows:

* Executing SQL statements
* Querying results and resultsets
* Handling SQL batches
* Other miscellaneous methods

The interface java.sql.statements defines

methods for executing different SQL statements like SELECT, UPDATE, INSERT, DELETE, and CREATE.

Public Resultset execute Query (string sql) throws SQLException

The following figure shows how the DriverManager, Driver, Connection, Statement, ResultSet classes are connected.

DriverManager

Driver

Driver

Layer

Application

Layer

Connection

Prepared Statement

Statement

Callable Statement

Result Set

Result Set

Result Set

**JAVA SERVER PAGES (JSP):**

**Introduction:**

Java Server Pages (JSP) technology enables you to mix regular, static HTML with dynamically generated content. You simply write the regular HTML in the normal manner, using familiar Web-page-building tools. You then enclose the code for the dynamic parts in special tags, most of which start with <% and end with %>.

**The need of JSP:**

Servlets are indeed useful, and JSP by no means makes them obsolete. However,

* It is hard to write and maintain the HTML.
* You cannot use standard HTML tools.
* The HTML is inaccessible to non-Java developers.

**Benefits of JSP:**

JSP provides the following benefits over servlets alone:

* It is easier to write and maintain the HTML: In this no extra backslashes, no double quotes, and no lurking Java syntax.
* You can use standard Web-site development tools:

We use Macromedia Dreamweaver for most of the JSP pages. Even HTML tools that know nothing about JSP can used because they simply ignore the JSP tags.

* You can divide up your development team:

The Java programmers can work on the dynamic code. The Web developers can concatenate on the representation layer. On large projects, this division is very important. Depending on the size of your team and the complexity of your project, you can enforce a weaker or stronger separation between the static HTML and the dynamic content.

**Creating template text:**

A large percentage of our JSP document consists of static text known as template text. In almost all respects, this HTML looks just likes normal HTML follows all the same syntax rules, and simply “passed through” to that client by the servlet created to handle the page. Not only does the HTML look normal, it can be created by whatever tools you already are using for building Web pages.

There are two minor exceptions to the “template text passed through” rule. First, if you want to have <% 0r %> in the out port, you need to put <\% or %\> in the template text. Second, if you want a common to appear in the JSP page but not in the resultant document,

<%-- JSP Comment -- %>

HTML comments of the form:

<!—HTML Comment -->

are passed through to the client normally.

**Types of JSP scrolling elements:**

JSP scripting elements allow you to insert Java code into the servlet that will be generated from the JSP page. There are three forms:

1. **Expressions** of the form <%=Java Expression %>, which are evaluated and inserted into the servlet’s output.
2. **Sciptlets** of the form <%Java code %>, which are inserted into the servlet’s\_jspService method (called by service).
3. **Declarations** of the form<%! Field/Method Declaration %>, which are inserted into the body of the servlet class, outside any existing methods.

**Using JSP Expressions:**

A JSP element is used to insert values directly into the output. It has the following form:

<%= Java Expression %>

The expression is evaluated, converted to a string, and inserted in the page. This evaluation is performed at runtime (when the page is requested) and thus has full access to the information about the request For example, the following shows the date/time that the page was requested.

Current time: <%=new java.util.Date () %>

**Predefined variables:**

To simplify expressions we can use a number of predefined variables (or “implicit objects”). The specialty of these variables is that, the system simple tells what names it will use for the local variables in \_jspService.The most important ones of these are:

* **request**, the HttpServletRequest.
* **response**, the HttpServletResponse.
* **session,** the HttpSession associated with the request
* **out,** the writer used to send output to clients.
* **application,** the ServletContext. This is a data structure shared by all servlets and JSP pages in the web application and is good for storing shared data.

Here is an example:

Your hostname: <%= **request.**getRemoteHost () %>

**Comparing servlets to JSP pages:**

JSP works best when the structure of the HTML page is fixed but the values at various places need to be computed dynamically. If the structure of the page is dynamic, JSP is less beneficial. Some times servlets are better in such a case. If the page consists of binary data or has little static content, servlets are clearly superior. Sometimes the answer is neither servlets nor JSP alone, but rather a combination of both.

**Writing scriptlets:**

If you want to do something more complex than output the value of a simple expression .JSP scriptlets let you insert arbitrary code into the servlet’s \_jspService method. Scriptlets have the following form:

<% Java code %>

Scriptlets have access to the same automatically defined variables as do expressions (request, response, session, out , etc ) .So for example you want to explicitly send output of the resultant page , you could use the out variable , as in the following example:

<%

String queryData = request.getQueryString ();

out.println (“Attached GET data: “+ queryData);

%>

**Scriptlet Examples:**

As an example of code that is too complex for a JSP expression alone, a JSP page that uses the bgColor request parameter to set the background color of the page .Simply using

<BODY BGCOLOR=”<%= request.getParameter (“bgcolor”) %> “>

would violate the cardinal rule of reading form data.

**Using declarations:**

A JSP declaration lets you define methods or fields that get inserted into the main body of the servlet class .A declaration has the following form:

<%! Field or Method Definition %>

Since declarations do not generate output, they are normally used in conjunction with JSP expressions or scriptlets. In principle, JSP declarations can contain field (instance variable) definitions, method definitions, inner class definitions, or even static initializer blocks: anything that is legal to put inside a class definition but outside any existing methods. In practice declarations almost always contain field or method definitions.

We should not use JSP declarations to override the standard servlet life cycle methods. The servlet into which the JSP page gets translated already makes use of these methods. There is no need for declarations to gain access to service, doget, or dopost, since calls to service are automatically dispatched to \_jspService , which is where code resulting from expressions and scriptlets is put. However for initialization and cleanup, we can use jspInit and jspDestroy- the standard init and destroy methods are guaranteed to call these methods in the servlets that come from JSP.

**Jakarta Tomcat:**

Tomcat is the Servlet/JSP container. Tomcat implements the Servlet 2.4 and JavaServer Pages 2.0 specification. It also includes many additional features that make it a useful platform for developing and deploying web applications and web services.

**Terminology:**

Context – a Context is a web application.

$CATALINA\_HOME – This represents the root of Tomcat installation.

**Directions and files:**

**/bin** – Startup, shutdown, and other scripts. The \*.sh files (for Unix systems) are functional duplicates of the \*.bat files (for Windows systems). Since the Win32 command-line lacks certain functionality, there are some additional files in here.

**/conf –** Configuration files and related DTDs. The most important file in here is server.xml. It is the main configuration file for the container.

**/logs –** Log files are here by default.

**/webapps –** This is where webapps go\

**Installation:**

Tomcat will operate under any Java Development Kit (JDK) environment that provides a JDK 1.2 (also known as Java2 Standard Edition, or J2SE) or later platform. JDK is needed so that servlets, other classes, and JSP pages can be compiled.

**Deployment directions for default web applications:**

**HTML and JSP Files**

* Main Location

$CATALINA\_HOME/webapps/ROOT

* Corresponding URLs.

http://host/SomeFile.html

http://host/SomeFile.jsp

* More Specific Location (Arbitrary Subdirectory).

$CATALINA\_HOME/webapps/ROOT/SomeDirectory

* Corresponding URLs

http://host/SomeDirectory/SomeFile.html

http://host/SomeDirectory/SomeFile.jsp

**Individual Servlet and Utility Class Files**

* Main Location (Classes without Packages).

$CATALINA\_HOME/webapps/ROOT/WEB-INF/classes

* Corresponding URL (Servlets).

http://host/servlet/ServletName

* More Specific Location (Classes in Packages).

$CATALINA\_HOME/webapps/ROOT/WEB-INF/classes/packageName

* Corresponding URL (Servlets in Packages).

http://host/servlet/packageName.ServletName

**Servlet and Utility Class Files Bundled in JAR Files**

* Location

$CATALINA\_HOME/webapps/ROOT/WEB-INF/lib

* Corresponding URLs (Servlets)

http://host/servlet/ServletName

http://host/servlet/packageName.ServletName

**XAMPP:**

XAMPP:

XAMPP stands for Cross-Platform (X), Apache (A), MariaDB (M), PHP (P) and Perl (P).

It is simply a web server if we want to make a website or designing and make a working website then XAMPP is useful .it gives an environment of hiw server works.

1. It contains apache, mysql, filezilla servers by which we can use them and helps us in login and logout sessions, cookies we gives a good help in websites

2. Also, it has wordpress feature by which it contains many themes of websites which are popular and we can use them to make a website without using so much php coding ,HTML, CSS etc.

3.How to use it: 1. if we are working on mysql then we just on the server of mysql and go to php admin page.

4.To work on php based web pages we just on the server and then, code on a notepad by using php pages.

**7.SOFTWARE TESTING**

**What do you mean by software testing?**

Testing involves operation of a system or application under controlled conditions and evaluating the results. The controlled conditions should include both normal and abnormal conditions. Testing should intentionally attempt to make things go wrong to determine if things happen when they shouldn't or things don't happen when they should. It is oriented to 'detection'.

|  |
| --- |
| spacer |

**7.1 Unit Testing:**

Unit testing is a software development process in which the smallest testable parts of an application, called units, are individually and independently scrutinized for proper operation. Unit testing is often automated but it can also be done manually. This testing mode is a component of Extreme Programming (XP), a pragmatic method of software development that takes a meticulous approach to building a product by means of continual testing and revision.

Unit tests are written from a programmer's perspective. They ensure that a particular method of a class successfully performs a set of specific tasks. Each test confirms that a method produces the expected output when given a known input.

**7.2 Performance Testing:**

Performance testing is the process of determining the speed or effectiveness of a computer, network, software program or device. This process can involve quantitative tests done in a lab, such as measuring the response time or the number of MIPS (millions of instructions per second) at which a system functions. Qualitative attributes such as

Reliability, scalability and interoperability may also be evaluated. Performance testing is often done in conjunction with stress testing.

Performance testing can verify that a system meets the specifications claimed by its manufacturer or vendor. The process can compare two or more devices or programs in terms of parameters such as speed, data transfer rate, bandwidth, throughput, efficiency or reliability.

Performance testing can also be used as a diagnostic aid in locating communications bottlenecks. Often a system will work much better if a problem is resolved at a single point or in a single component. For example, even the fastest computer will function poorly on today's Web if the connection occurs at only 40 to 50 Kbps (kilobits per second).

**7.3 Integration Testing:**

Integration testing, also known as integration and testing (I&T), is a software development process which program units are combined and tested as groups in multiple ways. In this context, a unit is defined as the smallest testable part of an application. Integration testing can expose problems with the interfaces among program components before trouble occurs in real-world program execution. Integration testing is a component of Extreme Programming (XP), a pragmatic method of software development that takes a meticulous approach to building a product by means of continual testing and revision.

**7.4 Test cases:**

**Test case for Login form:**

**Test case for User Registration form:**

**Test case3:**

**Test case for Change Password:**

When the old password does not match with the new password ,then this results in displaying an error message as “ OLD PASSWORD DOES NOT MATCH WITH THE NEW PASSWORD”.

**Test case 4:**

**Test case for Forget Password:**

When a user forgets his password he is asked to enter Login name, ZIP code, Mobile number. If these are matched with the already stored ones then user will get his Original password.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Module** | **Functionality** | **Test Case** | **Expected Results** | **Actual Results** | **Result** | **Priority** |
| User | Login Usecase | 1. Navigate To Www.Sample.Com 2. 2.Click On Submit Button Without Entering Username and Password | A Validation Should Be As Below “Please Enter Valid Username & Password” | A Validation Has Been Populated As Expected | Pass | High |
|  |  | 1. aNavigate To Www.Sample.Com 2. 2. Click On Submit Button With Out Filling Password And With Valid Username Test UsernameField | A Validation Should Be As Below “Please Enter Valid Password Or Password Field Can Not Be Empty “ | A Validation Is Shown As Expected | Pass | High |
|  |  | 1. NNavigate To Www.Sample.Com 2. Enter Both Username And Password Wrong And Hit Enter | A Validation Shown As Below “The Username Entered Is Wrong” | A Validation Is Shown As Expected | Pass | High |
|  |  | 1. Navigate To Www.Sample.Com 2. Enter Validate Username And Password And Click On Submit | Validate Username And Password In DataBase And Once If They Correct Then Show The Main Page | Main Page/ Home Page Has Been Displayed | Pass | High |

**SCREENSHOTS**

**8.CONCLUSION**

Biometric has its unique advantages over conventional password and token-based security system, as evidenced by its increased adoption (e.g., on Android and iOS devices). In this paper, we introduced a biometric-based mechanism to authenticate a user seeking to access services and computational resources from a remote location. Our proposed approach allows one to generate a private key from a fingerprint biometric reveals, as it is possible to generate the same key from a fingerprint of a user with 95.12% accuracy. Our proposed session key generation approach using two biometric data does not require any prior information to be shared. A comparison of our approach with other similar authentication protocols reveals that our protocol is more resilient to several known attacks. Future research includes exploring other biometric traits and also multi-modal biometrics for other sensitive applications (e.g., in national security matters).

**9. REFERENCES**

[1] C. Neuman, S. Hartman, K. Raeburn, “The kerberos network authentication service (v5),” RFC 4120, 2005.

[2] “OAuth Protocol.” [Online]. Available: <http://www.oauth.net/>

[3] “OpenID Protocol.” [Online]. Available: <http://openid.net/>

[4] G. Wettstein, J. Grosen, and E. Rodriguez, “IDFusion: An open architecture for Kerberos based authorization,” Proc. AFS and Kerberos Best Practices Workshop, June 2006.

[5] A. Kehne, J. Schonwalder, and H. Langendorfer, “A nonce-based protocol for multiple authentications,” ACM SIGOPS Operating System Review, vol. 26, no. 4, pp. 84–89, 1992.

[6] B. Neuman and S. Stubblebine, “A note on the use of timestamps as nonces,” Oper. Syst. Rev., vol. 27, no. 2, pp. 10–14, 1993.

[7] J. Astorga, E. Jacob, M. Huarte, and M. Higuero, “Ladon : endto-end authorisation support for resource-deprived environments,” IET Infomration Security, vol. 6, no. 2, pp. 93–101, 2012.

[8] S. Zhu, S. Setia, and S. Jajodia, “LEAP: efficient security mechanisms for large-scale distributed sensor networks,” Washington D.C., USA, October 2003, pp. 62–72.

[9] A. Perrig, R. Szewczyk, D. Tygar, V. Wen, and D. Culler, “SPINS: security protocols for sensor networks,” ACM Wireless Networking, vol. 8, no. 5, pp. 521–534, 2002.

[10] P. Kaijser, T. Parker, and D. Pinkas, “SESAME: The solution to security for open distributed systems,” Computer Communications, vol. 17, no. 7, pp. 501–518, 1994.

[11] G. Wettstein, J. Grosen, and E. Rodriguez, “IDFusion: An open architecture for Kerberos based authorization,” Proc. AFS and Kerberos Best Practices Workshop, June 2006.

[12] M. Walla, “Kerberos explained,” Windows 2000 Advantage Magazine, 2000. [13] Q. Jiang, J. Ma, X. Lu, and Y. Tian, “An efficient two-factor user authentication scheme with unlinkability for wireless sensor networks,” Peer-to-Peer Networking and Applications, vol. 8, no. 6, pp. 1070–1081, 2015. [14] O. Althobaiti, M. Al-Rodhaan, and A. Al-Dhelaan, “An efficient biometric authentication protocol for wireless sensor networks,” International Journal of Distributed Sensor Networks, vol. 2013, pp. 1–13, 2013, Article ID 407971, http://dx.doi.org/ 10.1155/2013/407971. [15] K. Xue, C. Ma, P. Hong, and R. Ding, “A temporal-credential-based mutual authentication and key agreement scheme for wireless sensor networks,” Journal of Network and Computer Applications, vol. 36, no. 1, pp. 316 – 323, 2013.

[16] M. Turkanovic, B. Brumen, and M. Holbl, “A novel user authentication and key agreement scheme for heterogeneous ad hoc wireless sensor networks, based on the internet of things notion,” Ad Hoc Networks, vol. 20, pp. 96 – 112, 2014. [17] M. Park, H. Kim, and S. Lee, “Privacy Preserving Biometric-Based User Authentication Protocol Using Smart Cards,” in 17th International Conference on Computational Science and Engineering, Chengdu, China, 2014, pp. 1541–1544. [18] P. K. Dhillon and S. Kalra, “A lightweight biometrics based remote user authentication scheme for IoT services,” Journal of Information Security and Applications, vol. 34, pp. 255 – 270, 2017.

[19] S. D. Kaul and A. K. Awasthi, “Security Enhancement of an Improved Remote User Authentication Scheme with Key Agreement,” Wireless Personal Communications, vol. 89, no. 2, pp. 621–637, 2016.

[20] D. Kang, J. Jung, H. Kim, Y. Lee, and D. Won, “Efficient and Secure Biometric-Based User Authenticated Key Agreement Scheme with Anonymity,” Security and Communication Networks, vol. 2018, pp. 1–14, 2018, Article ID 9046064, <https://doi.org/10.1155/2018/9046064>.

[21] D. Dolev and A. C. Yao, “On the security of public key protocols,” IEEE Transactions on Information Theory, vol. 29, no. 2, pp. 198–208, 1983.

[22] A. K. Das, “A secure and robust temporal credential-based three-factor user authentication scheme for wireless sensor networks,” Peer-to-Peer Networking and Applications, vol. 9, no. 1, pp. 223–244, 2016.

[23] “A secure and effective biometric-based user authentication scheme for wireless sensor networks using smart card and fuzzy extractor,” International Journal of Communication Systems, vol. 30, no. 1, pp. 1–25, 2017.

[24] C. T. Li, C. Y. Weng, and C. C. Lee, “An advanced temporal credentialbased security scheme with mutual authentication and key agreement for wireless sensor networks,” Sensors, vol. 13, no. 8, pp. 9589–9603, 2013.

[25] D. He, N. Kumar, and N. Chilamkurti, “A secure temporal-credentialbased mutual authentication and key agreement scheme for wireless sensor networks,” in International Symposium on Wireless and pervasive Computing (ISWPC), Taipei, Taiwan, 2013, pp. 1–6.

[26] M. Turkanovic and M. Holbl, “An improved dynamic password-based user authentication scheme for hierarchical wireless sensor networks,” ELEKTRONIKA IR ELEKTROTECHNIKA, vol. 19, no. 6, pp. 109 – 116, 2013. [27] R. Amin and G. P. Biswas, “A secure light weight scheme for user authentication and key agreement in multi-gateway based wireless sensor networks,” Ad Hoc Networks, vol. 36, pp. 58–80, 2016.

[28] C.-C. Chang and N.-T. Nguyen, “An Untraceable Biometric-Based Multi-server Authenticated Key Agreement Protocol with Revocation,” Wireless Personal Communications, vol. 90, no. 4, pp. 1695–1715, 2016.

[29] Z. Xia, C. Yuan, R. Lv, X. Sun, N. N. Xiong, and Y. Shi, “A Novel Weber Local Binary Descriptor for Fingerprint Liveness Detection,” IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2018, doi: 10.1109/TSMC.2018.2874281.

[30] C. Yuan, X. Sun, and Q. M. J. Wu, “Difference co-occurrence matrix using BP neural network for fingerprint liveness detection,” Soft Computing, vol. 23, no. 13, pp. 5157–5169, 2019.

[31] W. Yang, S. Wang, J. Hu, G. Zheng, and C. Valli, “Security and Accuracy of Fingerprint-Based Biometrics: A Review,” Symmetry, vol. 11, no. 2, 2019. [Online]. Available: <https://www.mdpi.com/2073-8994/11/2/141>

[32] X. Huang, X. Chen, J. Li, Y. Xiang, and L. Xu, “Further Observations on Smart-Card-Based Password-Authenticated Key Agreement in Distributed Systems,” IEEE Transactions on Parallel and Distributed Systems, vol. 25, no. 7, pp. 1767–1775, 2014.