SECURE CRYPTO BIOMETRIC SYSTEM FOR CLOUD COMPUTING

A project report submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Submitted by

P.SRI HARSHINI	(206N1A05B2)
A.S.K.CHANAKYA	(206N1A0566)
J.DEVI PRANEETHA	(206N1A0582)
K.SUNDARESWAR VARMA	(206N1A0585)

Under the esteemed guidance of

Mr. S. PRABHUDAS

Associate Professor & HOD

SRINIVASA INSTITUTE OF ENGINEERING AND TECHNOLOGY

(UGC – Autonomous Institution)

Approved by AICTE,Permanently affiliated to JNTUK Kakinada;ISO9001:2015 Certified Institute accredited NAAC with A grade ; Recognized by UGC Section2 (f) &2 (b) NH- 216Cheyyeru (v),Amalapuram-533222 Affiliated to JNTUK, Kakinada, A.P

2023-24

SRINIVASA INSTITUTE OF ENGINEERING AND TECHNOLOGY

(UGC – Autonomous Institution)

(Approved by AICTE, Permanently affiliated to JNTUK, Kakinada, ISO 9001: 2015 certified Institution)
(Accredited by NAAC with 'A' Grade; Recognised by UGC under sections 2(f) & 12(B))
NH-216, Amalapuram-Kakinada Highway, Cheyyeru (V), AMALAPURAM -533216.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project work entitled "**SECURE CRYPTO BIOMETRIC SYSTEM FOR CLOUD COMPUTING**" is bonafide work oF **P.SRI HARSHINI (206N1A05B2), A.S.K.CHANAKYA(206N1A0566), J.DEVIPRANEETHA (206N1A0582), and K.SUNDARESWAR VARMA (206N1A0585)** of IV B. Tech in Computer Science and Engineering, Srinivasa Institute of Engineering and Technology, Amalapuram Permanently affiliated to Jawaharlal Nehru Technological University Kakinada during the academic year 2023-2024 in partial fulfillment for the award of Bachelor of Technology in Computer Science and Engineering.

Mr. S. PRABHUDAS, M. Tech, Ph.D	**Mr. S. PRABHUDAS, M. Tech, Ph.D**
Head of the Department	**Internal Guide**

ACKNOWLEDGEMENTS

We express our sincere gratitude to our esteemed institute "**Srinivasa Institute of Engineering and Technology**", which has provided us with an opportunity to fill full the most cherished desire to reach our goal.

We owe our project to **MR. S. PRABHUDAS, Associate Professor & HOD**, Computer Science and Engineering Department, who has been our project guide. we sincerely thank him for the support and guidelines which was given to us as, without which we would not have made this effort of ours a success.

We express our deep-hearted thanks to **S. PRABHUDAS**, our beloved **Head of the Department** for being helpful in providing us with his valuable advice and timely guidance.

We would like to thank the principal **Dr. M. SRINIVASA KUMAR, Dean- IQAC Dr. B RATNA RAJU,** Dean-Academics **Dr. S JAYA LAKSHMI** and **MANAGEMENT** of **"Srinivasa Institute of Engineering and Technology**" for providing us with the requisite facilities to carry out our project in campus.

Our deep-hearted thanks to all the **FACULTY MEMBERS** of our department for their valuable based imparting of theory and practical subjects, which we have put into use in our project. We are also indebted to **THE NON-TEACHING STAFF** for their co-operation.

We would like to thank our **FRIENDS AND FAMILY MEMBERS** for their help and support in making our project a success.

P.SRI HARSHINI
A.S.K.CHANAKYA
J.DEVI PRANEETHA
K.SUNDARESWAR VARMA

CONTENTS

LIST OF FIGURES

ABSTRACT

Cloud computing has achieved maturity, and there is a heterogeneous group of providers and cloud-based services. However, significant attention remains focused on security concerns. In many cases, security and privacy issues are a significant barrier to user acceptance of cloud computing systems and the advantages these offer with respect to previous systems. Biometric technologies are becoming the key aspect of a wide range of secure identification and personal verification solutions, but in a cloud computing environment they present some problems related to the management of biometric data, due to privacy regulations and the need to trust cloud providers. To overcome those problems in this paper, we propose a crypto-biometric system applied to cloud computing in which no private biometric data are exposed.

1. INTRODUCTION

Cloud computing is a trend in application architecture and development, as well as a new Business model. The success of many service providers, with Amazon as a remarkable example, has demonstrated that the model can be applied to a wide variety of solutions, covering the different levels defined in the cloud paradigm (SaaS, PaaS and IaaS). We can consider that cloud computing is at a mature stage, although there remain some limitations and challenges.

Cloud computing brings import and benefits for organizations that out source data, applications, and infrastructure, at the cost of delegating data control. The information is processed in computers that the users do not own, operate, or manage. In this scenario, the user does not know how the provider handles the information, and therefore a high level of trust is needed. The lack of control over physical and logical aspects of the system imposes profound changes in security and privacy procedures.

Currently there is even a lack of service level agreements between providers and users regarding security. Our research focuses on adequate security mechanisms that should potentially meet the legal requirements of traditional systems

In recent years, intensere search work has been devoted to biometrics, and their applications to security have become more evident. The combination of cloud computing and biometric technology opens new research and implementation opportunities to protect user data in the cloud. However, biometric templates must remain secure even for cloud services providers given the privacy-sensitive nature of biometric data. This requisite is more important than in the case of alphanumerical passwords due to the impossibility of changing user information in the same way

1.1 Aim of the Project

The aim of a secure crypto biometric system for cloud computing would likely be to develop a robust authentication and encryption system that integrates biometric data (like fingerprints or iris scans) with cryptographic techniques to ensure secure access and data protection in cloud environments. This would enhance security by adding an additional layer of identity verification beyond traditional passwords or tokens.

Furthermore, the project aims to overcome the limitations of existing biometric systems by integrating them seamlessly into cloud computing infrastructures. This involves developing algorithms and protocols that facilitate the secure transmission and storage of biometric data while ensuring compliance with privacy regulations and standards. By harnessing the scalability and flexibility of cloud technologies, the system can accommodate a wide range of applications and users while maintaining robust security measures.

Ultimately, the project seeks to establish a paradigm shift in cloud security, where biometric authentication becomes a standard practice for accessing sensitive data and services. By combining the strengths of cryptography and biometrics, the system aims to provide a holistic approach to security that mitigates risks associated with unauthorized access, data breaches, and identity theft in cloud computing environments. Through rigorous testing and validation, the project endeavors to deliver a reliable and efficient solution that meets the evolving needs of organizations and individuals alike.

1.2 Motivation

Overcome the barriers to user acceptance caused by security and privacy issues in cloud computing by employing a crypto-biometric system that preserves data privacy and fosters trust. Traditional authentication methods, such as passwords and tokens, are susceptible to various attacks, posing significant risks to sensitive information stored in the cloud. Integrating biometric authentication into cloud computing offers a compelling solution to these security challenges. Biometric data, such as fingerprints, iris scans, or facial recognition, provides a unique and inherent identifier for individuals, significantly enhancing the accuracy and reliability of identity verification. By coupling biometric authentication with cryptographic techniques, organizations can establish a robust security framework that safeguards access to sensitive data and resources in the cloud.

1.3 Objectives

- A secure solution for managing biometric data in the cloud, ensuring compliance with privacy regulations and establishing trust in cloud providers
- Another key objective is to streamline the user authentication experience within cloud environments. By integrating biometric authentication, such as fingerprints or facial recognition, the system aims to provide a seamless and user-friendly method for accessing cloud services. This not only improves user experience but also mitigates the risks associated with weak passwords or stolen credentials.
- Furthermore, the objective extends to ensuring compliance with regulatory requirements and industry standards governing data protection and privacy. By incorporating cryptographic protocols, the system aims to encrypt sensitive data

during transmission and storage, thereby safeguarding it from unauthorized interception or disclosure. This compliance-centric approach fosters trust among users and organizations, reinforcing the integrity and security of cloud computing ecosystems.

- Overall, the objective of a secure crypto biometric system for cloud computing is to establish a robust framework that enhances security, improves user experience, and ensures compliance with regulatory mandates.

1.4 Secure crypto biometric system for cloud computing

1. Biometric Data Collection: Collect biometric data from users using secure sensors or devices
2. Feature Extraction: Extract unique features from the biometric data, such as fingerprints, iris patterns, or facial feature
3. Encryption: Encrypt the biometric data using strong cryptographic algorithms to protect it during transmission and storage in the cloud
4. Biometric Template Protection: Store biometric templates securely using techniques like cryptographic hashing or template transformation to prevent reverse engineering
5. Secure Communication: Ensure secure communication between the client and the cloud server using protocols like TLS/SSL to prevent interception or tampering of data.
6. Authentication Protocols: Implement robust authentication protocols to verify the identity of users based on their biometric data, such as biometric match-on-server or match-on-card.
7. Access Control: Enforce access control mechanisms to restrict access to biometric data and authentication processes only to authorized parties.
8. Multi-Factor Authentication: Consider combining biometric authentication with other factors like passwords or tokens to enhance security.
9. Continuous Monitoring: Monitor system activity and detect any unauthorized access or suspicious behavior in real-time.
10. Regular Updates and Patching: Keep the system up-to-date with the latest security patches and updates to mitigate vulnerabilities.

1.5 Organization of Thesis

Chapter-1: A brief introduction about our project contains objectives, motivation and methodologies.

Chapter-2: Provides the literature survey based on several studies and previous researcher papers and magazines are referred. It contains authors name and their proposed methodologies.

Chapter-3: Different existing Methodologies are explained for existed and their drawbacks on previous researchers and Proposed system to overcome the drawbacks using PCA and UBM algorithms is presented, System architecture and system execution process for results and discussion, Data flow diagram and their UML diagrams in detailed.

Chapter-4: Operating Environment i.e, software and hardware requirements for this project.

Chapter-5: System Development Environment Technology used to Develop Project Python and Framework TKinter, database and other libraries.

Chapter-6: System testing i.e. unit testing and integrating testing and how to implement.

Chapter-7: Project Sample code and Main Code are accessed in this project.

Chapter-8: Results and Outputs. The code Execution screenshots, and process will be shown and explained.

Chapter-9: Conclusion and future scope will be evaluated on this project.

2. LITERATURE SURVEY

Achieving Secure, Scalable, and Fine-Grained Data Access Control in CloudComputing (S.Yu,C. Wang, K.Ren)

Cloud computing is a new computing paradigm where computing resources are provided over the Internet. However, it poses challenges for data security and access control, especially when sensitive data is shared on un trusted cloud servers. Existing solutions that use cryptography introduce computational overhead for key distribution and management, limiting scalability. This paper addresses this challenge by enforcing access policies based on data attributes and delegating computation tasks to un trusted servers without revealing data contents. The proposed scheme combines attribute-based encryption, proxy re-encryption, and lazy re-encryption, ensuring efficiency, security, user access privilege confidentiality, and user key accountability. Extensive analysis proves its effectiveness under existing security models.

Generating Cancelable Fingerprint Templates (N. Ratha,S.ChikkerurJ)

Biometrics-based authentication systems offer convenience compared to traditional password or token-based methods. However, they raise privacy concerns due to the permanence of biometric identifiers. Once compromised, a biometric is lost forever and can be used across multiple applications to track a user. This paper presents methods to generate multiple cancelable identifiers from fingerprint images, allowing users to have as many biometric identifiers as needed. These identifiers can be canceled and replaced if compromised. The performance of different algorithms, such as Cartesian, polar, and surface folding transformations, is compared. Through experiments, it is shown that revocability and prevention of cross-matching biometric databases can be achieved. The transforms are also proven to be noninvertible, ensuring the original biometric cannot be recovered from a transformed version.

Cancelable templates for sequence-based biometrics with application to online signature recognition (E. Maiorana, P. Campisi)

Biometric technologies for automatic people recognition have gained popularity, but security and privacy concerns hinder their widespread adoption. This paper presents a solution called Bio Convolving, which ensures security and renewability of biometric templates. The approach employs non invertible transformations on biometric templates represented by sequences, generating multiple transformed versions. The transformed templates are computationally hard to reverse back to the original data. The proposed approach is demonstrated using an online signature recognition system, employing a hidden Markov model-based matching strategy. Extensive experiments using the MCYT signature database validate the effectiveness of the protected and renewable biometric templates, albeit with a slight impact on authentication performance.

Combining crypto biometrics effectively (F.Hao,R. Anderson)

We propose the first practical and secure way to integrate there is biometric in to cryptographic applications. A repeatable binary string, which we call a biometric key, is generated reliably from genuine iris codes. A well-known difficulty has been how to cope with the 10 to 20 percent of error bits with in an iris code and derive an error-freekey. To solve this problem, we carefully studied the error patterns within iris codes and devised a two-layer error correction technique that combines Had amard and Reed-Solomon codes. The key is generated from a subject's iris image with the aidofauxiliary error-correction data, which do not reveal the key and can be saved in a tamper-resistant token, such as a smartcard. The reproduction of the key depends on two factors: the iris biometric and the token. The attacker has to procure both of them to compromise the key.

We evaluated our technique using iris samples from70 different eyes, with 10 samples from each eye. We found that an error-free key can be reproduced reliably from genuine iris codes with a 99.5 percent success rate. We can generate up to 140 bits of biometric key, more than enough for128-bit AES. The extraction of a repeatable binary string from biometrics opens new possible applications, where a strong binding is required between a person and cryptographic operations. For example, it is possible to identify individuals without maintaining a central database of biometric templates, to which privacy objections might be raised.

A Posteriori Estimation for Multivariate Gaussian Mixture Observations of Markov Chains(J-L Gauvain, C-H Lee)

In this paper, a framework for maximum a posteriori (MAP) estimation of hidden Markov models (HMM) is presented. Three key issues of MAP estimation, namely, the choice of prior distribution family, the specification of the parameters of prior densities, and the evaluation of the MAP estimates, are addressed. Using HMM's with Gaussian mixtures to observation densities as an example, it is assumed that the prior densities for the HMM parameters can be adequately represented as a product of Dirichlet and normal Wishart densities. The classical maximum ikelihood estimation algorithms, namely, the forward-backward algorithm and the segmental k-means algorithm, are expanded, and MAP estimation formulas are developed. Prior density estimation issues are discussed for two classes of applications/spl minus/parameters moothing and model adaptation/splminus/ and some experimental results are given illustrating the practical interest of this approach. Because of its adaptive nature, Bayesian learning is shown to serve as a unified approach for a wide range of speech recognition application.

3. METHODOLOGY

3.1 Existing System

Now-a-days all organizations or techie peoples are using cloud services to store and manage their business data as this cloud services provide heavy computing resources in cheaper cost but this advantage leads to an issue called data security as data store at cloud is away from user's hand and can be misuse by cloud service provider or hacker.

Disadvantages:

1. Data susceptibility to breaches.
2. Limited user data control.
3. Dependence on cloud providers.
4. Potential for unauthorized access.
5. Risk of data loss.
6. Reliance on internet connectivity

3.2 Proposed System:

The proposed system integrates cutting-edge cryptographic techniques with biometric authentication for enhanced security in cloud computing environments. Leveraging advanced encryption algorithms and secure key management protocols, sensitive data is protected both in transit and at rest. Biometric authentication adds an extra layer of security, ensuring that only authorized users can access the system. By utilizing cloud infrastructure, the system offers scalability and flexibility while maintaining robust security measures. Continuous monitoring and anomaly detection mechanisms further fortify the system against potential threats, ensuring the integrity and confidentiality of user data in the cloud environment. Overall, the integration of cryptography and biometrics in cloud computing provides a highly secure and efficient platform for handling sensitive information. To overcome from this problem author is employing Crypto-Biometric which will convert user biometric data in to encrypted format and only genuine verification system can only verify such data so if hacker get crypto-biometric data then he will not aware of decoder key so he cannot verify himself.

In proposed using biometric database as Universal Background Model (UBM)whose features will be extracted using HMM(hidden markov model) algorithm and this algorithm will use Baum-Welch and LBG clustering technique to extract features from biometric template. Extracted features will get through PCA algorithm for features selection and then selected features will been coded using Private Key. Encoded features will get train with GMM algorithm mand above features encoding process is called as Fuzzy Commitment Template Protection.

Advantages of Proposed System

1. Enhanced data privacy assurance.
2. Improved user trust levels.
3. Strength end security measures.
4. Efficient biometric data management.
5. Minimized risk of data exposure
6. Perfect user verification process

Universal Background Model: A Universal Background Model (UBM) is a model used in a biometric verification system to represent general, person-independent feature characteristics to be compared against a model of person-specific feature characteristics when making an accept or reject decision. For example, in a speaker verification system, the UBM is a speaker-independent Gaussian Mixture Model (GMM) trained with speech samples from a large set of speakers to represent general speech characteristics. Using a speaker-specific GMM trained with speech samples from a particular enrolled speaker, a likelihood-ratio test for an unknown speech sample can be formed between the match score of the speaker-specific model and the UBM. The UBM may also be used while training the speaker-specific model by acting as a the prior model in Maximum A Posteriori (MAP) parameter estimation.

Algorithm:

Here's a step-by-step algorithm for creating a UBM:

1.Collecting Speech Data: Gather a large amount of speech data from various sources. This data should ideally cover a wide range of speakers, languages, and speaking styles.

2. Feature Extraction: Extract acoustic features from the speech data. Common features used in speaker recognition systems include Mel- Frequency Cepstral Coefficients (MFCCs), which represent the spectral characteristics of the speech signal.

3.Feature Normalization: Normalize the extracted features to ensure that they are consistent across different speakers and recording conditions. This normalization step helps in reducing variability due to factors such as recording equipment differences and speaking distance from the microphone.

4. Gaussian Mixture Model (GMM) Training: Train a GMM using Expectation-Maximization (EM) algorithm on the normalized feature vectors. This involves modeling the distribution of the features using a mixture of Gaussian distributions. The number of Gaussian components in the mixture is a parameter that needs to be determined empirically.

5. Universal Background Model (UBM) Construction: Once the GMM is trained, it serves astheUBM.Thismodelcapturesthestatisticalpropertiesoftheentiredatasetwithoutspecific information about individual speakers. Each Gaussian component in the GMM represents a cluster of feature vectors in the feature space.

6. Finalization: Optionally, you can refine the UBM further by adapting it to better represent the specific characteristics of the target speaker population. This can be done using techniques Like Maximum Likelihood Linear Regression (MLLR) or feature space Maximum a Posteriori (MAP) adaptation.

7. Deployment: The UBM is then used as are ferencemodel in a speaker recognition system. During the identification or verification process, the similarity between a speaker's voice and the UBM is computed to determine the likelihood of the speaker being the same as there ference or a different one.

Principal Component Analysis (PCA):

Principal component analysis (PCA) is a dimensionality reduction and machine learning method used to simplify a large data set into a smaller set while still maintaining significant patterns and trends. Reducing the number of variables of a data set naturally comes at the expense of accuracy, but the trick in dimensionality reduction is to trade a little accuracy for simplicity. Because smaller data sets are easier to explore and visualize, and thus make analyzing data points much easier and faster for machine learning algorithms without extraneous variables to process

Algorithm:

Standardize the Data: If the features of your dataset are on different scales, it’s essential to standardize them (subtract the mean and divide by the standard deviation).

1. **Compute the Covariance Matrix:** Calculate the covariance matrix for the standardized datas
2. **Compute Eigen vectors and Eigen values**: Find the eigenvectors and eigen values of the covariance matrix. The eigenvectors represent the directions of maximum variance, and the corresponding eigenvalues indicate the magnitude of variance along those directions.

3. **Sort Eigen vectors by Eigen values:** Sort the eigenvectors based on their corresponding eigen values in descending order.

4. **Choose Principal Components**: Select the top eigenvectors (principal components) where is the desired dimensionality of the reduced dataset.

5. Transform the Data: Multiply the original standardized data by the selected principal components to obtain the new, lower-dimensional representation of the data

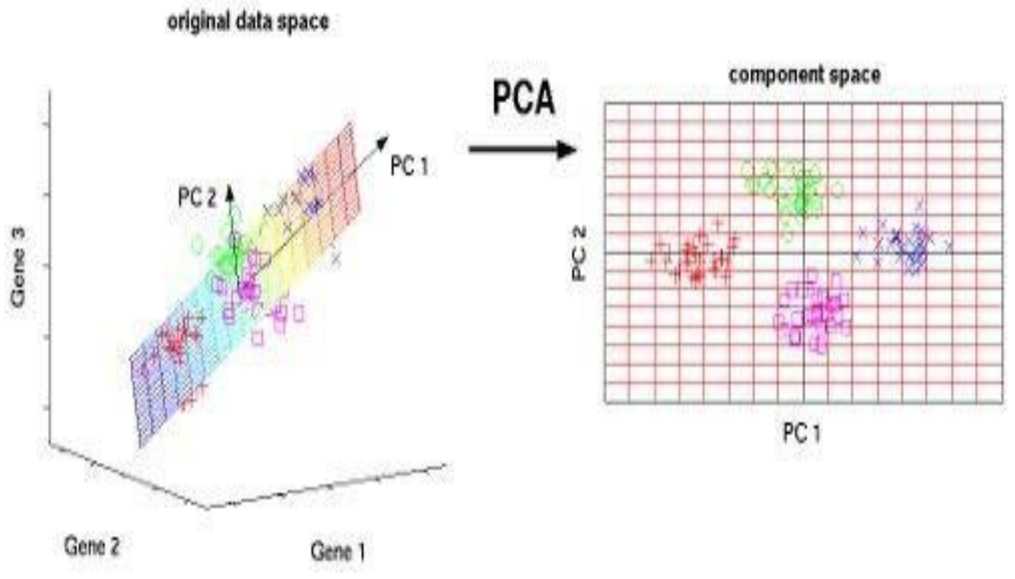


Figure 1 Principal Component Analysis Diagram

.Module Description

User Interface Setup:

The project utilizes Tkinter, a Python library for creating graphical user interfaces (GUI). The main window titled "Secure crypto-biometric system for cloud computing" is initialized with specific dimensions (1300x1200).

Library Imports:

Import necessary libraries including Tkinter for GUI, Open CV (cv2) for image processing, NumPy for numerical computations, and other libraries such as PCA, Gaussian Mixture, and AES encryption.

Key Generation and Encryption Setup:

The project employs ECC (Elliptic Curve Cryptography) for encryption and decryption.

ECC keys (public and private) are generated using the generate Key() function.

AES (Advanced Encryption Standard) keys are generated using PBKDF2 (Password-Based Key Derivation Function 2).

Database Upload and Label Extraction:

Users can upload biometric data bases containing images.

Biometric labels are extracted from the uploaded data base to identify individuals.

Feature Extraction:

The system extracts features from biometric templates in the database.

Features are extracted using Open CV and resized to a standard size (28x28pixels).

Extracted features are stored in NumPy arrays.

Feature Selection and Encoding:

PCA (Principal Component Analysis) is applied for feature selection to reduced imensionality.

The selected features are encoded using a BCH (Bose-Chaudhuri-Hocquenghem) encoder for error detection and correction.

AES and ECC Encryption Time Graph:

Users can visualize the execution time for AES and ECC encryption using a bargraph.

Graphical User Interface(GUI):

The GUI provides buttons for various functionalities like uploading the database, feature extraction, feature selection, encoding, verification, and graph display.

Text boxes display the status and results of each operation.

AES and ECC Encryption Time Graph:

Users can visualize the execution time for AES and ECC encryption using a bar graph.

Graphical User Interface(GUI):

The GUI provides buttons for various functionalities like uploading the database, feature extraction, feature selection, encoding, verification, and graph display.

Text boxes display the status and results of each operation.

3.3 System Architecture

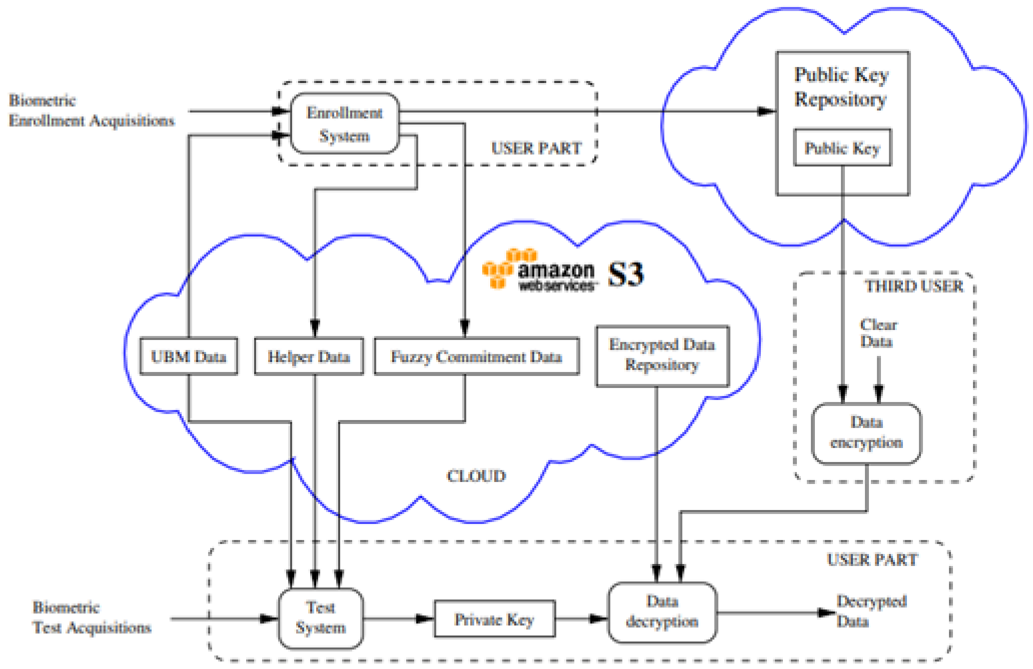


Figure 2 System Architecture

3.4 Data Flow Diagram

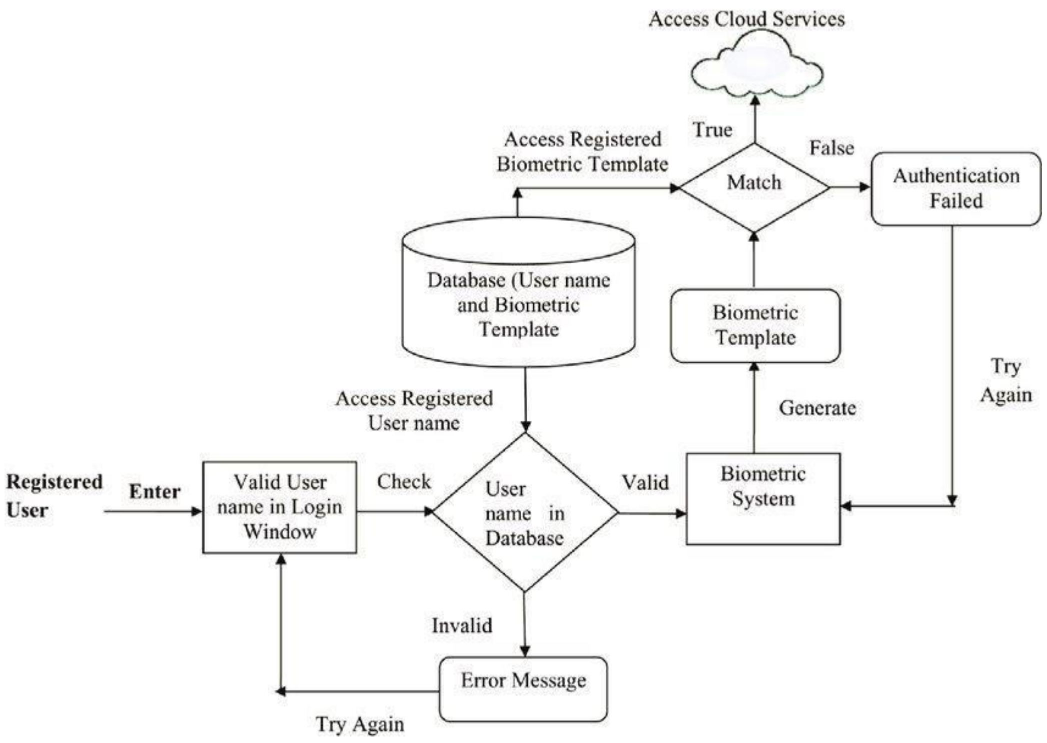


Figure 3 Data Flow Diagram

3.5 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. Initscurrent 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.

3.5.1 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 a use case), 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.

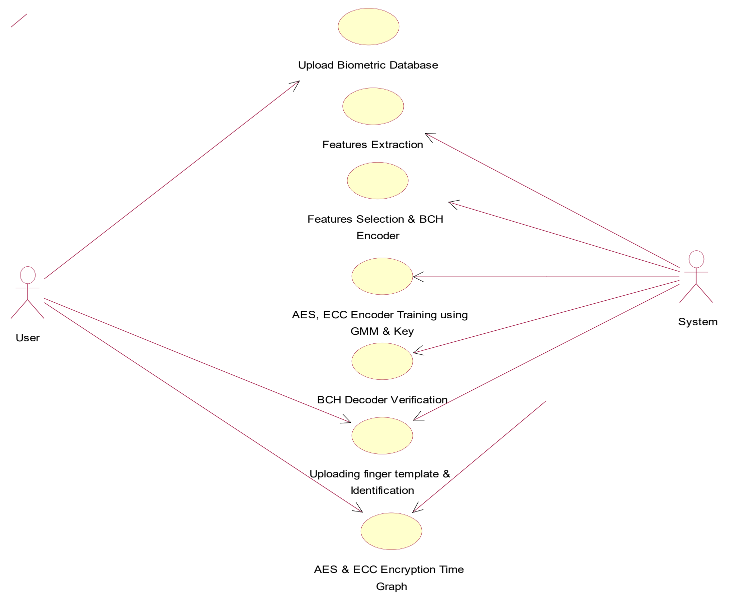

Upload Biometric Database
Features Extraction
Features Selection & BCH Encoder
AES, ECC Encoder Training using GMM & Key
BCH Decoder Verification
Uploading finger template & Identification
AES & ECC Encryption Time Graph
User
System

Figure 4 Class Diagram

3.5.2 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 diagram.

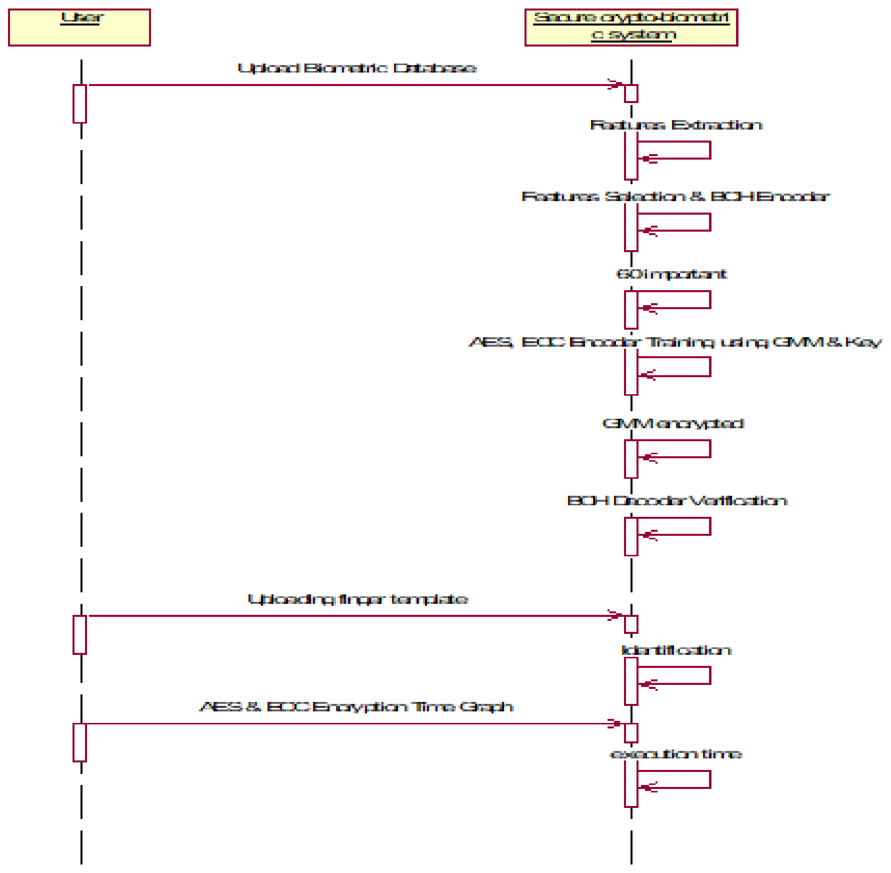

User
Secure crypto-biometri c system
Upload Biometric Database
Features Extraction
Features Selection & BCH Encoder
60 important
AES, ECC Encoder Training using GMM & Key
GMM encrypted
BCH Decoder Verification
Uploading finger template
Identification
AES & ECC Encryption Time Graph
execution time

Figure 5 Sequence Diagram

3.5.3 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.

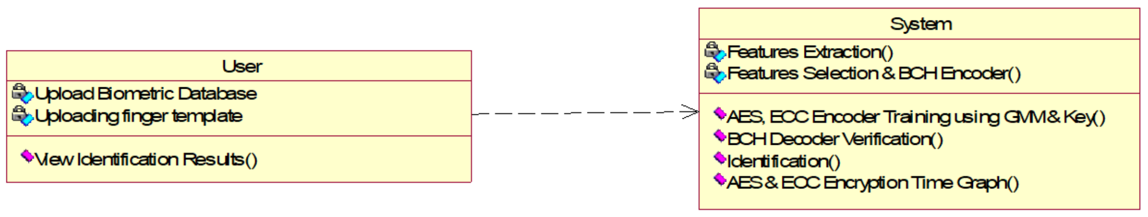

User
Upload Biometric Database
Uploading finger template
View Identification Results()
System
Features Extraction()
Features Selection & BCH Encoder()
AES, ECC Encoder Training using GMM & Key()
BCH Decoder Verification()
Identification()
AES & ECC Encryption Time Graph()

Figure 6 Class Diagram

3.5.4 Collaboration Diagram

The collaboration diagram is used to show the relationship between the objects in a system. Both the sequence and the collaboration diagrams represent the same information but differently. Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming. An object consists of several features. Multiple objects present in the system are connected to each other. The collaboration diagram, which is also known as a communication diagram, is used to portray the object's architecture in the system.

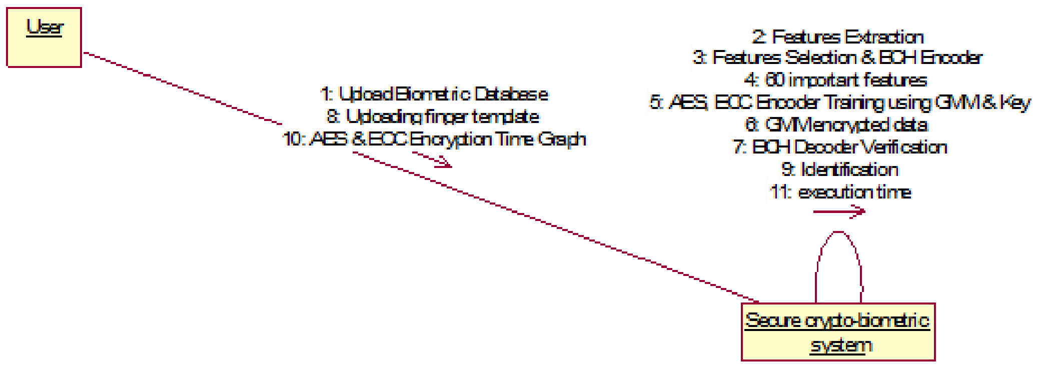

User
1: Upload Biometric Database
8: Uploading finger template
10: AES & ECC Encryption Time Graph
2: Features Extraction
3: Features Selection & BCH Encoder
4: 60 important features
5: AES, ECC Encoder Training using GMM & Key
6: GMMencrypted data
7: BCH Decoder Verification
9: Identification
11: execution time
Secure crypto-biometric system

Figure 7 Collaboration Diagram

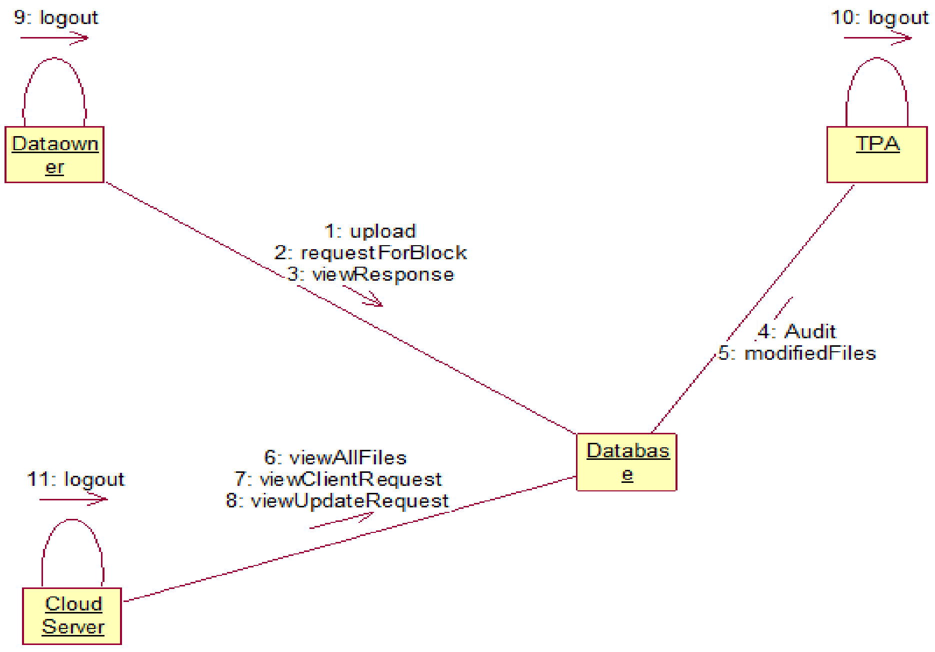


Figure 8 Activity Diagram

4. EXPERIMENTAL SPECIFICATIONS

4.1 HARDWARE REQIREMENTS

• System	:	Pentiumi3/i5.
• Hard Disk	:	500GB.
• Monitor	:	15’’LED
• Input Devices	:	Keyboard, Mouse
• Ram	:	4GB

4.2 SOFTWAREREQIREMENTS

Operating system	:	Windows 11/10.
Coding Language	:	PYTHON

5. SYSTEM DEVELOPMENT ENVIRONMENT

5.1 Python Technology

Python Program

- Python programs are composed of modules
- Modules contain statements
- Statements contain expressions
- Expressions create and process objects

Features of Python

Open source: Python is publicly available open source software, anyone can use source code that doesn't cost anything.

Easy-to-learn: Popular (scripting/extension) language, clear and easy syntax, no type declarations, automatic memory management, high-level data types and operations, design to read (more English like syntax) and write (shorter code compared to C, C++, and Java) fast.

High-level Language:

High-level language (closer to human) refers to the higher level of concept from machine language (for example assembly languages). Python is an example of a high-level language like C, C++, Perl, and Java with low-level optimization.

Portable:

High level languages are portable, which means they are able to run across all major hardware and software platforms with few or no change in source code. Python is portable and can be used on Linux, Windows, Macintosh, Solaris, FreeBSD, OS/2, Amiga, AROS, AS/400 and many more.

Object-Oriented: Python is a full-featured object-oriented programming language, with features such as classes, inheritance, objects, and overloading.

Python is Interactive:

Python has an inter active console where you get a Python prompt (command line) and interact with the interpreter directly to write and test your programs. This is useful for mathematical programming.

Interpreted: Python programs are interpreted, takes source code as input, and the n compiles (to portable byte-code) each statement and executes it immediately. No need to compiling or linking.

Python is often referred to as a "glue" language, meaning that it is capable to work in mixed- language environment. The Python interpreter is easily extended and can add anew built-in function or modules written in C/C++/Java code.

Libraries: Databases, web services, networking, numerical packages, graphical user interfaces, 3D graphics, others.

Support: Support from online Python community

History

The name Python was selected from "Monty Pythons Flying Circus" which was a British sketch comedy series created by the comedy group Monty Python and broadcast by the BBC from 1969 to 1974. Python was created in the early 1990s by Guido van Rossum at the National Research Institute for Mathematics and Computer Science in Netherlands.

Between 1991 and 2001 there are several versions released, current stable release is 3.2. In 2001 the Python Software Foundation (PSF) was formed, a non-profit organization created specifically to own Python-related Intellectual Property. ZOPE Corporation is a sponsoring member of the PSF.

All most all Python releases are Open Source. To see the details of release versions and licence

agreement of Python.

Python Environment

AIX	AROS	AS/400 (OS/400)	BeOS
MORPH OS	MS-DOS	OS/2	OS/390and z/OS
Palm OS	PlayStation and PSP	Psion	QNX
RISCOS	Series 60	Solaris	VMS
Windows C Eorpocket	HP-UX	LINUX	

Major uses of Python

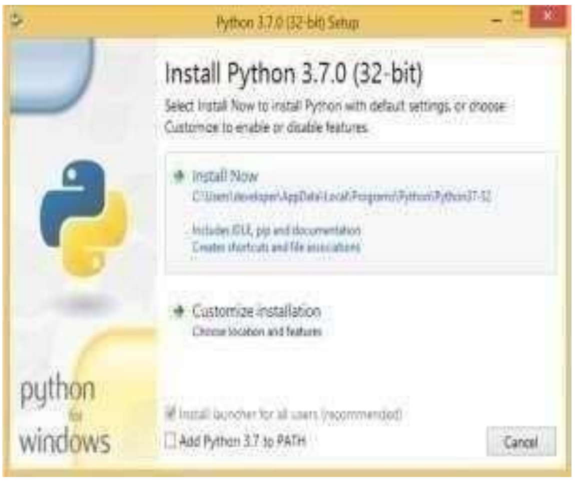
- System utilities (system admin tools, command line programs).
- Web Development.
- Graphical User Interfaces (Tkinter, gtk, Qt).

- Internet scripting.
- Embedded scripting.
- Database access and programming.
- Game programming.
- Rapid prototyping and development.
- Distributed programming

Installation on Windows

Visit the link https://www.python.org/downloads/ to download the latest release of Python. In this process, we will install Python 3.7.0 on our Windows operating system.

Double-click the executable file which is downloaded; the following window will open. Select Customize installation and proceed.



Python 3.7.0 (32-bit) Setup

Install Python 3.7.0 (32-bit)

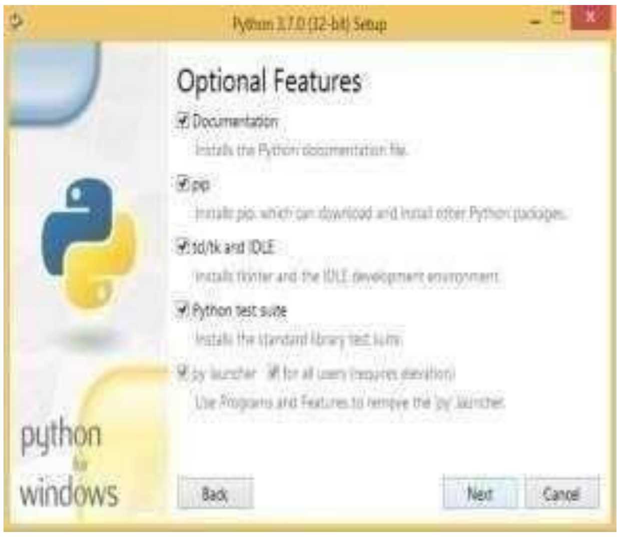
Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

Install Now
C:\Users\developer\AppData\Local\Programs\Python\Python37-32
Includes IDLE, pip and documentation
Creates shortcuts and file associations

Customize installation
Choose location and features

☑ Install launcher for all users (recommended)
☐ Add Python 3.7 to PATH

The following windows how 'install' the optional features. All the features need to be installed and are checked by default; we need to click next to continue.



Python 3.7.0 (32-bit) Setup

Optional Features

☑ Documentation
Installs the Python documentation file.

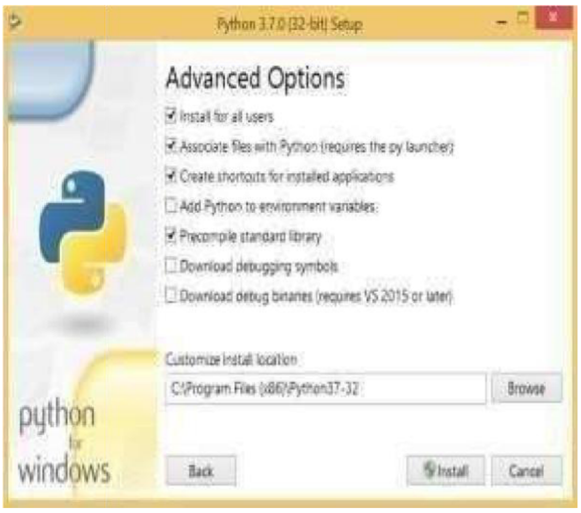
☑ pip
Installs pip, which can download and install other Python packages.

☑ td/tk and IDLE
Installs tkinter and the IDLE development environment.

☑ Python test suite
Installs the standard library test suite.

☑ py launcher ☑ for all users (requires elevation)
Use Programs and Features to remove the 'py' launcher.

The following window shows a list of advanced options. Check all the options which you want to install and click next. Here, we must notice that the first check-box (install for all users) must be checked.


Python 3.7.0 (32-bit) Setup

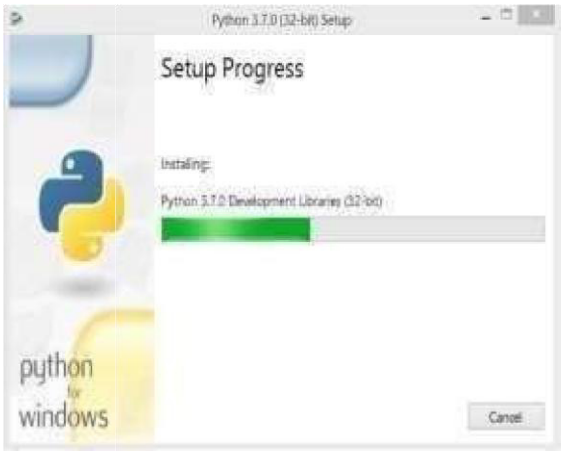
Advanced Options

- ✓ Install for all users
- ✓ Associate files with Python (requires the py launcher)
- ✓ Create shortcuts for installed applications
- ☐ Add Python to environment variables
- ✓ Precompile standard library
- ☐ Download debugging symbols
- ☐ Download debug binaries (requires VS 2015 or later)

Customize install location

C:\Program Files (x86)\Python37-32

Now, we are ready to install python-3.7. Let us install it.

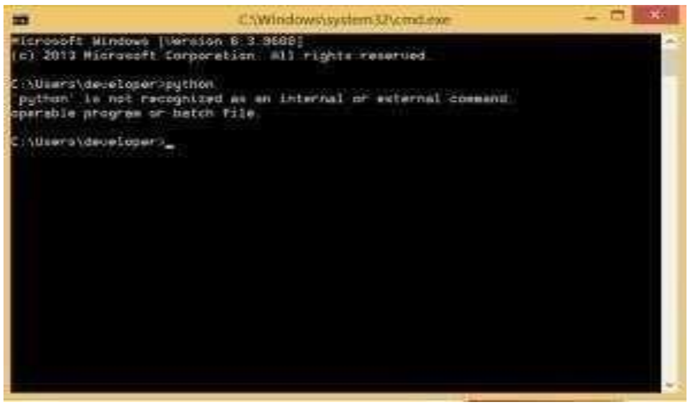

Python 3.7.0 (32-bit) Setup

Setup Progress

Installing:

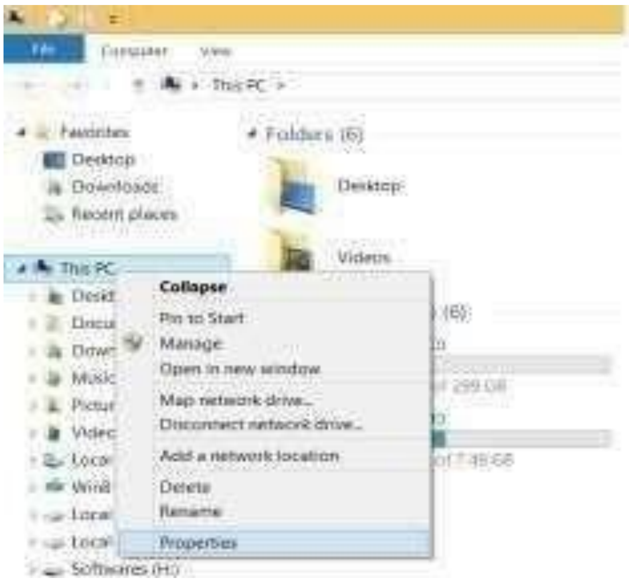
Python 3.7.0 Development Libraries (32-bit)

Now, try to run python on the command prompt. Type the command python in case of python2 or python3 in case of python3. It will show an error as given in the below image. It is because we haven't set the path.

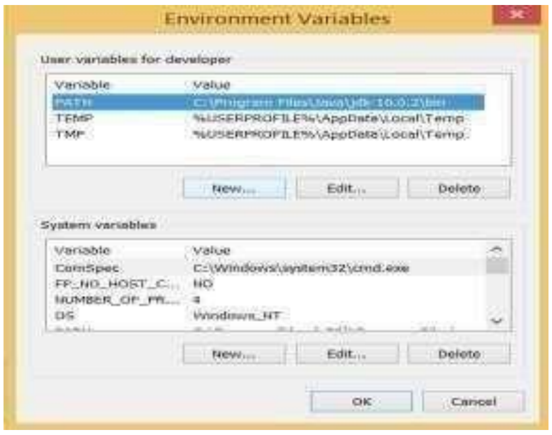


```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\developer>python
'python' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\developer>_
```

To set the path of python, we need to the right click on "my computer" and go to Properties →Advanced → Environment Variables.



Add the new path variable in the user variable section



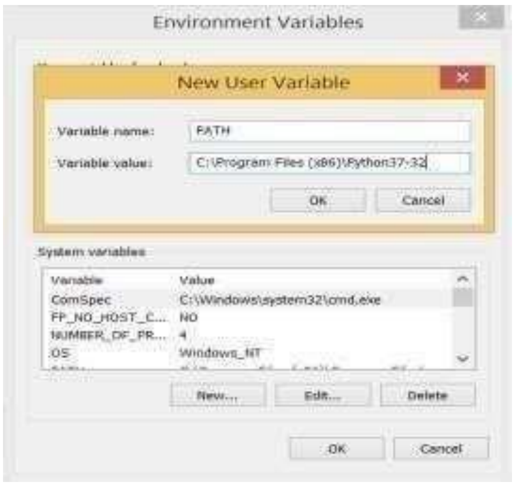
User variables for developer

Variable	Value
PATH	C:\Program Files\Java\jdk-10.0.2\bin
TEMP	%USERPROFILE%\AppData\Local\Temp
TMP	%USERPROFILE%\AppData\Local\Temp

System variables

Variable	Value
ComSpec	C:\Windows\system32\cmd.exe
FP_NO_HOST_C...	NO
NUMBER_OF_PR...	4
OS	Windows_NT

Type PATH as the variable name and set the path to the installation directory of the python shown in the below image.


Environment Variables

New User Variable

Variable name: PATH

Variable value: C:\Program Files (x86)\Python37-32

System variables

Variable	Value
ComSpec	C:\Windows\system32\cmd.exe
FP_NO_HOST_C...	NO
NUMBER_OF_PR...	4
OS	Windows_NT

Now, the path is set, we are ready to run python on our local system. Restart CMD and type python again. It will open the python interpreter shell where we can execute the python statements.

6. SYSTEM TESTING

Testing Methodologies

The following are the Testing Methodologies:

o Unit Testing.

o Integration Testing. o System Testing. o Functional Testing.

Unit Testing:

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

This System consists of 3 modules. Those are Reputation module, route discovery module, audit module. Each module is taken as unit and tested. Identified errors are corrected and executable unit are obtained.

Integrating Testing:

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

6.1 System Testing

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

Functional Testing:

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centered on the following items

Valid Input : identified classes of valid input must be accepted

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised

Output : identified classes of application outputs must be exercised.

Systems/ Procedures : interfacing systems or procedures must be invoked

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes

6.2 Types of Testing

Testing is a process, which reveals errors in the program. It is the major quality measure employed during software development. During testing, the program is executed with a set of conditions known as test cases and the output is evaluated to determine whether the program is performing as expected. In order to make sure that the system does not have errors, the different levels of testing strategies that are applied at differing phases of software development are:

Black Box Testing

In this strategy some test cases are generated as input conditions that fully execute all functional requirements for the program.

- Interface errors
- Errors in data structure or external database access
- Performance errors
- Initialization and termination errors.

In this testing only the output is checked for correctness. The logical flow of the data is not checked.

White Box Testing

In this testing, the test cases are generated on the logic of each module by drawing flow graphs of that module and logical decisions are tested on all the cases. It has been uses to generate the test cases in the following cases:

1. Guarantee that all independent paths have been executed.
2. Execute all logical decisions on their true and false sides
3. Execute all loops at their boundaries and within their operational
4. Execute internal data structures to ensure their validity.

System Testing and Implementation:

The purpose is to exercise the different parts of the module code to detect coding errors. After this the modules are gradually integrated into subsystems, which are then integrated themselves too eventually forming the entire system. During integration of module integration testing is performed. The goal of this is to detect designing errors, while focusing the interconnection between modules. After the system was put together, system testing is performed. Here the system is tested against the system requirements to see if all requirements were met and the system performs as specified by the requirements. Finally accepting testing is performed to demonstrate to the client for the operation of the system.

For the testing to be successful, proper selection of the test case is essential. There are two different approaches for selecting test case. The software or the module to be tested is treated as a black box, and the test cases are decided based on the specifications of the system or module. For this reason, this form of testing is also called "black box testing". The focus here is on testing the external behavior of the system. In structural testing the test cases are decided based on the logic of the module to be tested. A common approach here is to achieve some type of coverage of the statements in the code. The two forms of testing are complementary: one tests the external behavior, the other tests the internal structure.

Testing is an extremely critical and time-consuming activity. It requires proper planning of the overall testing process. Frequently the testing process starts with the test plan. This plan identifies all testing related activities that must be performed and specifies the schedule, allocates the resources, and specifies guidelines for testing. The test plan specifies conditions that should be tested; different units to be tested, and the manner in which the module will be integrated together. Then for different test unit, a test case specification document is produced, which lists all the different test cases, together with the expected outputs, that will be used for testing. During the testing of the unit the specified test cases are executed and the actual results are compared with the expected outputs. The final output of the testing phase is the testing report and the error report, or a set of such reports. Each test report contains a set of test cases and the result of executing the code with the test cases. The error report describes the errors encountered and the action taken to remove the error.

7. SAMPLE CODE

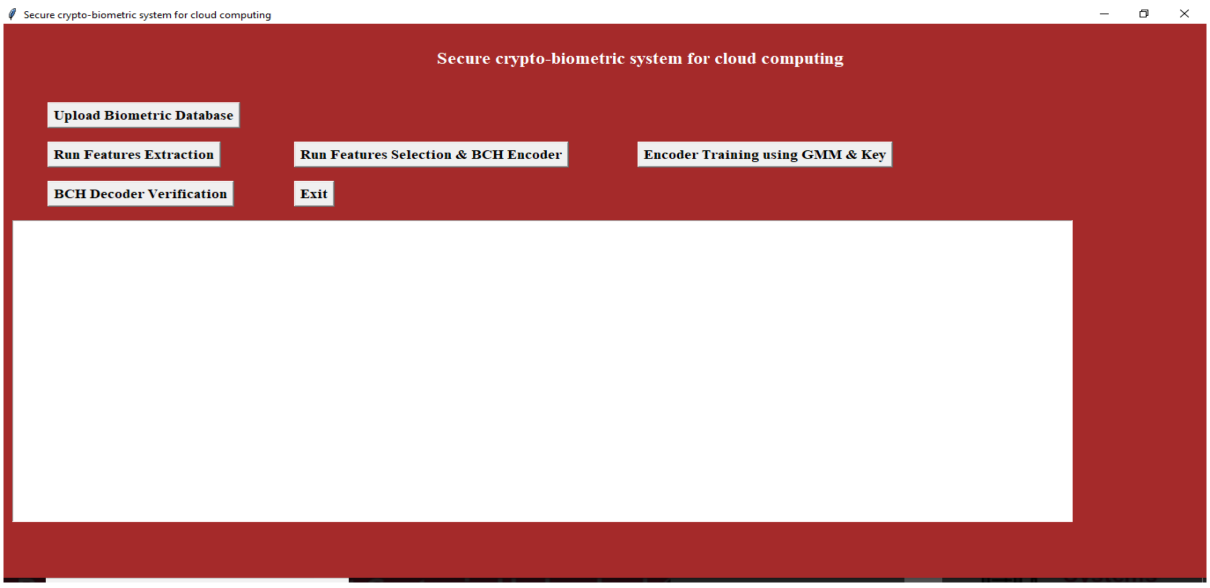
Secure crypto-biometric system for cloud computing

```
fromtkinterimportmessagebox from tkinter import *
fromtkinterimportsimpledialog import tkinter
fromtkinterimport filedialog
fromtkinter.filedialogimportaskopenfilename import os
importnumpyasnp import cv2
importmatplotlib.pyplotasplt
from sklearn.decomposition import PCAfromsklearn.mixtureimportGaussianMixture
from sklearn.metrics import accuracy_score import pickle
fromecies.utilsimportgenerate_eth_key,generate_key
from ecies import encrypt, decrypt #importing classes for ECC
encryptionimportpyaes,pbkdf2,binascii,os,secrets#importingclassesforAESasPYAES
import time
main=tkinter.Tk()
main.title("Securecrypto-biometricsystemforcloudcomputing")
main.geometry("1300x1200")
global filename, pathlabel globalX,Y,encoder,pca,gmm global labels
globalecc_publicKey,ecc_privateKey#definingpublicandprivatekeysvariablesforECC
globalaes_time, ecc_time
defECCEncrypt(obj):#ECCencryptionfunction enc = encrypt(ecc_publicKey, obj)
returnenc
defECCDecrypt(obj):#ECCdecryptionfunction dec = decrypt(ecc_privateKey, obj)
returndec
defgenerateKey():#functiontogenerateECCkeys global ecc_publicKey,ecc_privateKey
eth_k = generate_eth_key() ecc_private_key=eth_k.to_hex()
ecc_public_key=eth_k.public_key.to_hex() return ecc_private_key, ecc_public_key
defgetAesKey():#generatingkeywithPBKDF2forAES password = "s3cr3t*c0d3"
passwordSalt='76895'
key=pbkdf2.PBKDF2(password,passwordSalt).read(32) return key
defAesencrypt(plaintext):#AESdataencryption
aes = pyaes.AESModeOfOperationCTR(getAesKey(),
pyaes.Counter(31129547035000047302952433967654195398124239844566322884172l6
36 37846056248223))
ciphertext=aes.encrypt(plaintext) return ciphertext
defAesdecrypt(enc):#AESdatadecryption
aes = pyaes.AESModeOfOperationCTR(getAesKey(),
pyaes.Counter(3112954703500004730295243396765419539812423984456632288417216
36 37846056248223))
decrypted= aes.decrypt(enc)
returndecrypted
defreadLabels(path): global labels
forroot,dirs,directoryinos.walk(path): for j in range(len(directory)):
name=os.path.basename(root) if name not in labels:
labels.append(name)
defgetID(name): label = 0
foriinrange(len(labels)): if name == labels[i]:
label=i break
returnlabel
def uploadDatabase(): globalfilename,labels
labels = []
filename=filedialog.askdirectory(initialdir=".") pathlabel.config(text=filename)
text.delete('1.0', END) text.insert(END,filename+"loaded\n\n") readLabels(filename)
text.insert(END,"TotalpersonsbiometrictemplatesfoundinDatabase:
"+str(len(labels))+"\n\n")
text.insert(END,"PersonDetails\n\n") text.insert(END, str(labels))
def featuresExtraction(): global filename text.delete('1.0',END)
globalX,Y
ifos.path.exists("model/X.npy"): X = np.load("model/X.npy") Y =
np.load("model/Y.npy")
else:
X= []
Y= []
forroot,dirs,directoryinos.walk(filename): for j in range(len(directory)):
name = os.path.basename(root)if'Thumbs.db'notindirectory[j]:
img=cv2.imread(root+"/"+directory[j],0) img = cv2.resize(img, (28,28))
label = getID(name) X.append(img.ravel()) Y.append(label) print(str(label)+""+name)
```

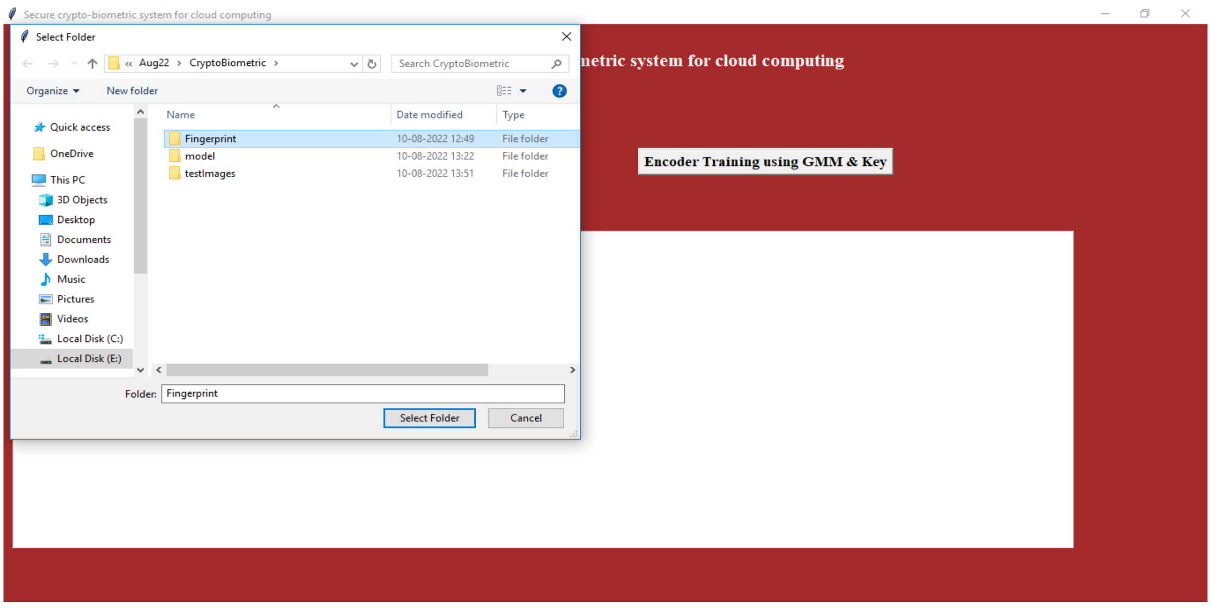
```
X=np.asarray(X) Y= np.asarray(Y)
X=X.astype('float32') X = X/255
np.save("model/X",X) np.save("model/Y",Y)
text.insert(END,"ExtractedFeaturesfromtemplates\n\n") text.insert(END,str(X))
def featuresSelection(): text.delete('1.0',END)
globalX,Y,pca,encoder
   text.insert(END,"TotalfeaturesavailableintemplatesbeforeapplyingPCAfeatures
selection: "+str(X.shape[1])+"\n\n")
pca=PCA(n_components=60) X = pca.fit_transform(X)
   text.insert(END,"TotalfeaturesavailableintemplatesafterapplyingPCAfeatures
selection: "+str(X.shape[1])+"\n\n")
Siritext.insert(END,"EncoderfeaturesafterencryptingwithKEY\n\n") encoder = []
foriinrange(len(X)): temp = []
for j in range(len(X[i])): temp.append(X[i,j]**2)
   encoder.append(temp) encoder=np.asarray(encoder) text.insert(END,str(encoder))
defrunGMMEncoding(): text.delete('1.0',END)
globalecc_publicKey,ecc_privateKey global aes_time, ecc_time
globalencoder,Y,gmm
if os.path.exists('model/gmm.txt'):
withopen('model/gmm.txt','rb')asfile: gmm = pickle.load(file)
   file.close() else:
gmm=GaussianMixture(n_components=10,max_iter=1000) gmm.fit(encoder, Y)
   #gmmistheobjectwhichisusedforverificationanditcontainsalltemplatesdetailsso GMM
has to get encrypted
start=time.time()
ecc_privateKey,ecc_publicKey=generateKey()#getting ECC keys
gmm=ECCEncrypt(pickle.dumps(gmm))#nowencryptingGMMusingECC gmm =
pickle.loads(ECCDecrypt(gmm))#now decrypting GMM using ECC end = time.time()
ecc_time=end -start#calculatingECCencryptionanddecryptiontime

#nowencryptingwithAES
start=time.time()#gettingAESstart time
gmm=Aesencrypt(pickle.dumps(gmm))#doingAESencryptiononGMM
encrypted_data=gmm[0:400] end = time.time()
```

```
aes_time = end - start #calculating AES encryption and decryption time
gmm=pickle.loads(Aesdecrypt(gmm))#doingAESdecryptiononGMM ecc_time =
ecc_time * 4
    text.insert(END,"Encodertraining&AES&ECCEncryptionprocesscompletedon
GMM\n\n")
text.insert(END,"TimetakenbyAES :"+str(aes_time)+"\n\n")
text.insert(END,"TimetakenbyECC:"+str(ecc_time)+"\n\n") text.insert(END,"Encrypted
Data\n\n") text.insert(END,str(encrypted_data))
def verification(): text.delete('1.0',END) global pca, gmm
filename=filedialog.askopenfilename(initialdir="testImages") img =
cv2.imread(filename,0)
img=cv2.resize(img,(28,28)) test = [] test.append(img.ravel())
test=np.asarray(test)
test=test.astype('float32') test = test/255
test=pca.transform(test) decoder = []
foriinrange(len(test)): temp = []
foriinrange(len(test)):
 temp = []
for j in range(len(test[i])): temp.append(test[i,j]**2)
    decoder.append(temp) decoder=np.asarray(decoder)
predict=gmm.predict(decoder)[0] img = cv2.imread(filename)
img=cv2.resize(img, (600,400))
    cv2.putText(img,'Biometrictemplatebelongstoperson:'+str(predict),(10,25),
cv2.FONT_HERSHEY_SIMPLEX,0.7, (255, 0, 0), 2)
cv2.imshow('Biometrictemplatebelongstoperson:'+str(predict),img) cv2.waitKey(0)
def graph():
global aes_time, ecc_time height=[aes_time,ecc_time]
bars=('AESExecutionTime','ECCExecutionTime') y_pos = np.arange(len(bars))
plt.bar(y_pos, height) plt.xticks(y_pos,bars)
plt.title("AES&ECCExecutionTimeGraph") plt.show()
def GUI():
globaltext,main,pathlabel font = ('times', 16, 'bold')
title=Label(main,text='Securecrypto-biometricsystemforcloudcomputing')
title.config(bg='brown', fg='white')
title.config(font=font) title.config(height=3,width=120) title.place(x=0,y=5)
font1=('times',13,'bold')
   uploadButton=Button(main,text="UploadBiometricDatabase",
command=uploadDatabase)
uploadButton.place(x=50,y=100) uploadButton.config(font=font1)
pathlabel = Label(main) pathlabel.config(bg='brown',fg='white')
pathlabel.config(font=font1)
pathlabel.place(x=460,y=100)
   extractionButton=Button(main,text="RunFeaturesExtraction",
command=featuresExtraction)
extractionButton.place(x=50,y=150) extractionButton.config(font=font1)
   selectionButton=Button(main,text="RunFeaturesSelection&BCHEncoder",
command=featuresSelection)
selectionButton.place(x=330,y=150) selectionButton.config(font=font1)
encodingButton=Button(main,text="AES,ECCEncoderTrainingusingGMM&Key",
command=runGMMEncoding)
encodingButton.place(x=720,y=150) encodingButton.config(font=font1)

   verificationButton=Button(main,text="BCHDecoderVerification",
command=verification)
verificationButton.place(x=50,y=200) verificationButton.config(font=font1)
   graphButton=Button(main,text="AES&ECCEncryptionTimeGraph", command=graph)
graphButton.place(x=330,y=200) graphButton.config(font=font1)
font1 = ('times', 12, 'bold') text=Text(main,height=20,width=150) scroll=Scrollbar(text)
text.configure(yscrollcommand=scroll.set) text.place(x=10,y=250) text.config(font=font1)
main.config(bg='brown') main.mainloop()
if name______=="main": GUI()
```

8. RESULTS

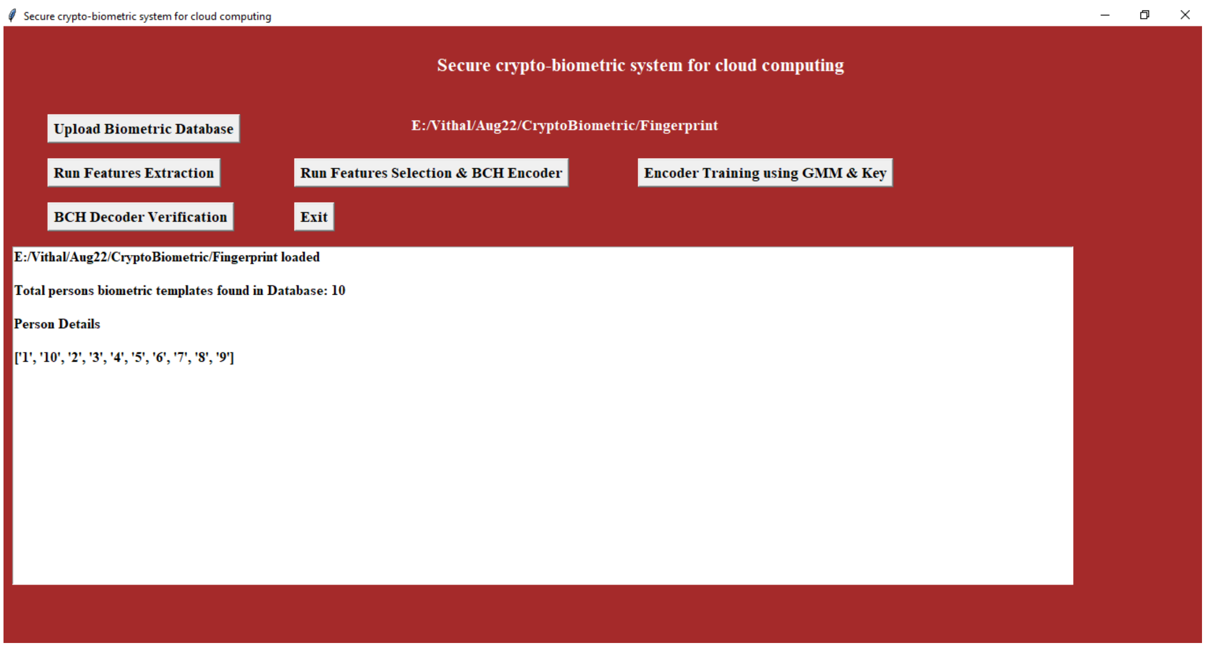

Secure crypto-biometric system for cloud computing
Secure crypto-biometric system for cloud computing
Upload Biometric Database
Run Features Extraction
Run Features Selection & BCH Encoder
Encoder Training using GMM & Key
BCH Decoder Verification
Exit

In above screen click on 'Upload Biometric Data base' button to upload database and get below screen.



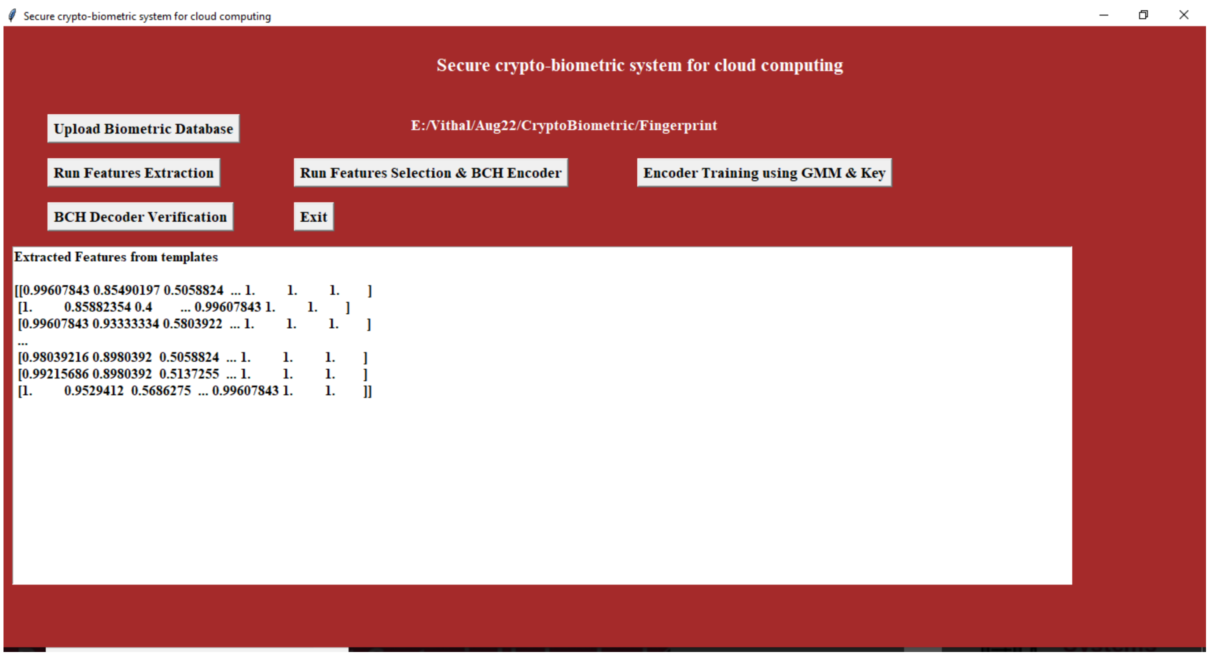
Name	Date modified	Type
Fingerprint	10-08-2022 12:49	File folder
model	10-08-2022 13:22	File folder
testImages	10-08-2022 13:51	File folder

In above screens electing and uploading finger print data base and then click on 'Select Folder' button to load database and get below output


Secure crypto-biometric system for cloud computing
Secure crypto-biometric system for cloud computing
Upload Biometric Database
E:/Vithal/Aug22/CryptoBiometric/Fingerprint
Run Features Extraction
Run Features Selection & BCH Encoder
Encoder Training using GMM & Key
BCH Decoder Verification
Exit

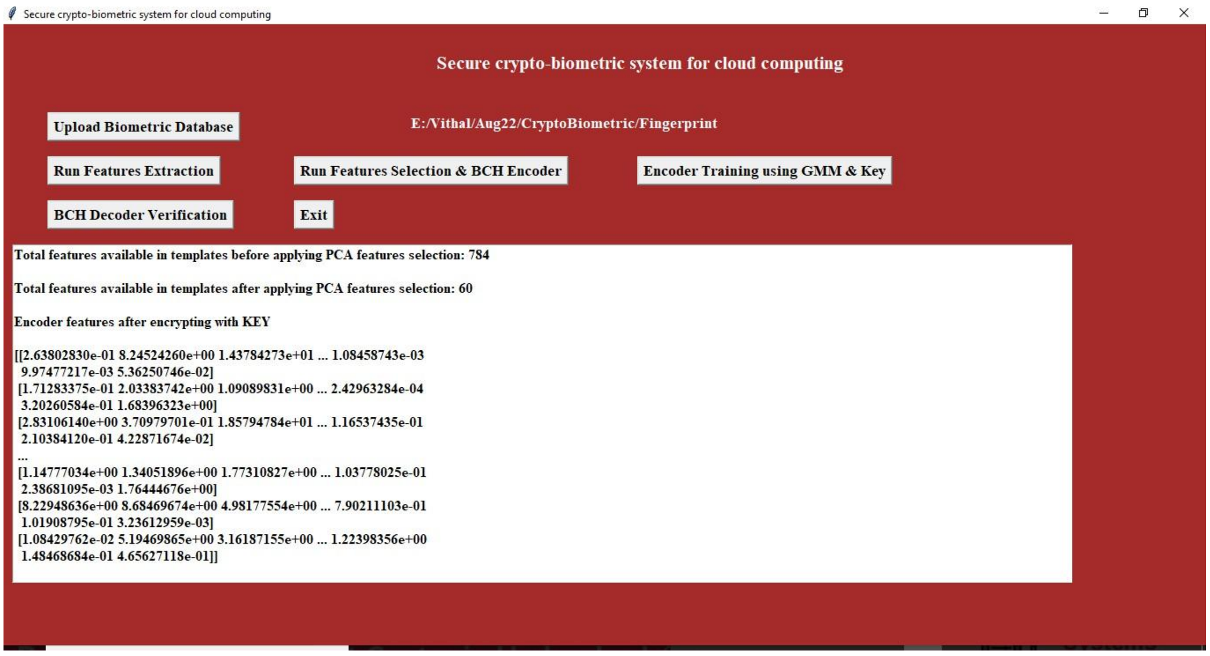
```
E:/Vithal/Aug22/CryptoBiometric/Fingerprint loaded

Total persons biometric templates found in Database: 10

Person Details

['1', '10', '2', '3', '4', '5', '6', '7', '8', '9']
```

In above screen database loaded and it contains details of 10 persons and now click on ‘Run Features Extraction’ button to extract features from database and get below output


Secure crypto-biometric system for cloud computing
Secure crypto-biometric system for cloud computing
Upload Biometric Database
E:/Vithal/Aug22/CryptoBiometric/Fingerprint
Run Features Extraction
Run Features Selection & BCH Encoder
Encoder Training using GMM & Key
BCH Decoder Verification
Exit

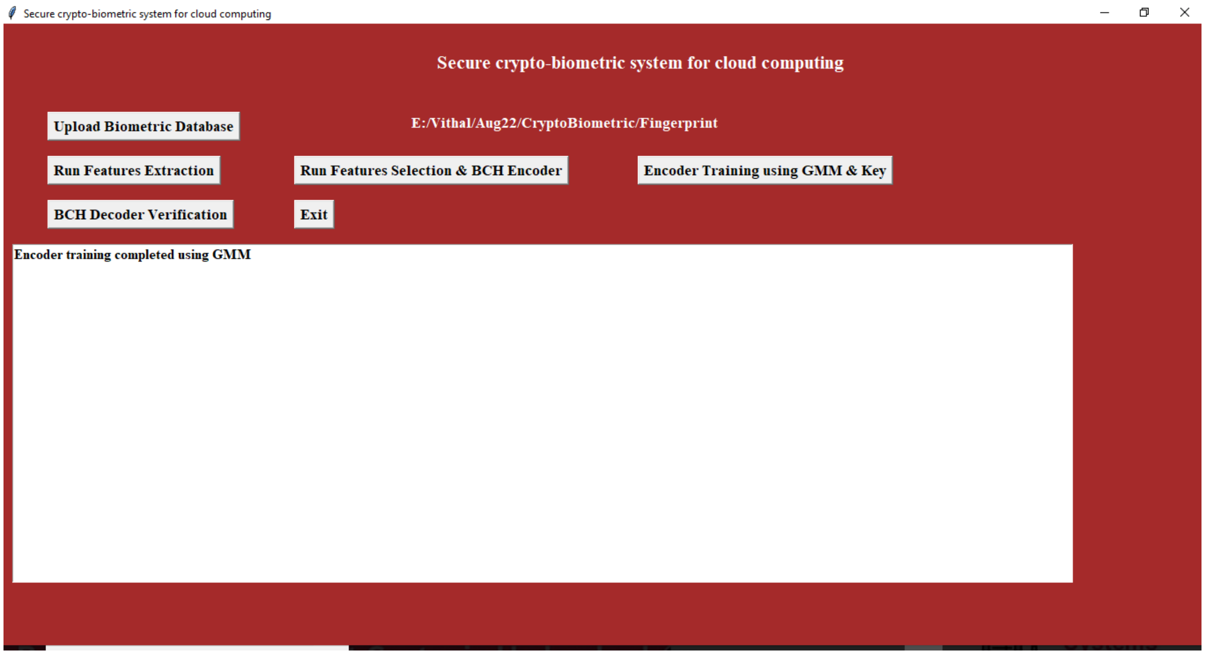
```
Extracted Features from templates

[[0.99607843 0.85490197 0.5058824  ... 1.         1.         1.        ]
 [1.         0.85882354 0.4        ... 0.99607843 1.         1.        ]
 [0.99607843 0.93333334 0.5803922  ... 1.         1.         1.        ]
 ...
 [0.98039216 0.8980392  0.5058824  ... 1.         1.         1.        ]
 [0.99215686 0.8980392  0.5137255  ... 1.         1.         1.        ]
 [1.         0.9529412  0.5686275  ... 0.99607843 1.         1.        ]]
```

In above screen showing some extracted features from database and now click on‘Run Features Selection & BCH Encoder’ button to select features and then apply encoding


Secure crypto-biometric system for cloud computing
Secure crypto-biometric system for cloud computing
Upload Biometric Database
E:/Vithal/Aug22/CryptoBiometric/Fingerprint
Run Features Extraction
Run Features Selection & BCH Encoder
Encoder Training using GMM & Key
BCH Decoder Verification
Exit

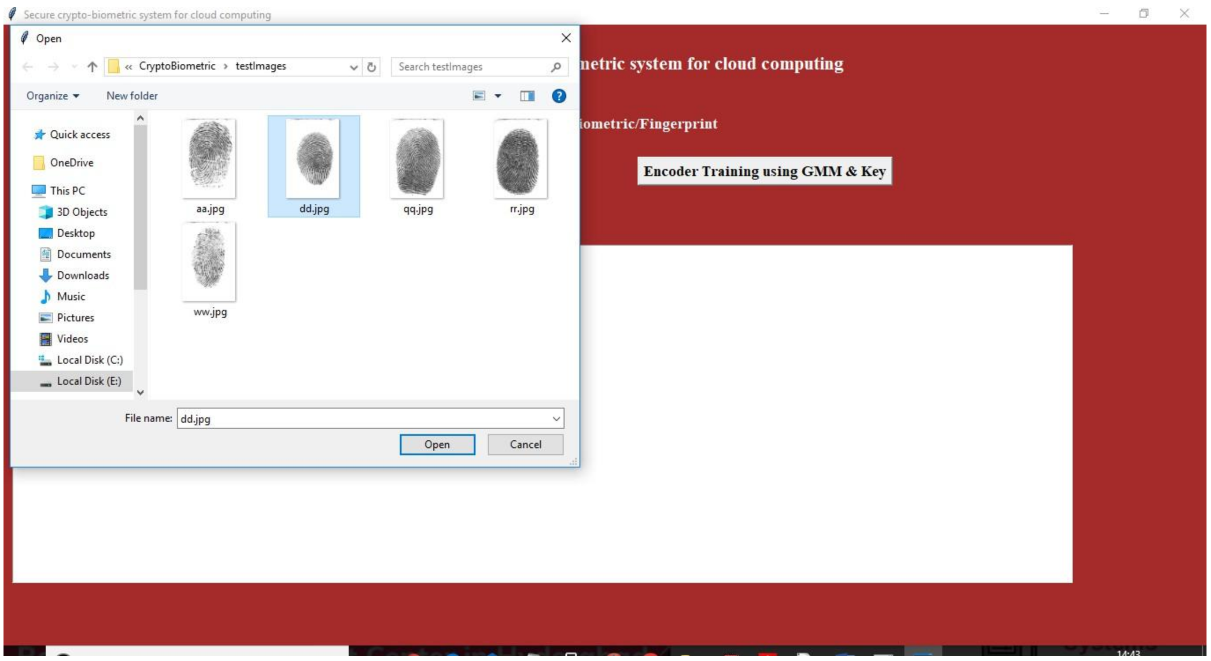
```
Total features available in templates before applying PCA features selection: 784

Total features available in templates after applying PCA features selection: 60

Encoder features after encrypting with KEY

[[2.63802830e-01 8.24524260e+00 1.43784273e+01 ... 1.08458743e-03
  9.97477217e-03 5.36250746e-02]
 [1.71283375e-01 2.03383742e+00 1.09089831e+00 ... 2.42963284e-04
  3.20260584e-01 1.68396323e+00]
 [2.83106140e+00 3.70979701e-01 1.85794784e+01 ... 1.16537435e-01
  2.10384120e-01 4.22871674e-02]
 ...
 [1.14777034e+00 1.34051896e+00 1.77310827e+00 ... 1.03778025e-01
  2.38681095e-03 1.76444676e+00]
 [8.22948636e+00 8.68469674e+00 4.98177554e+00 ... 7.90211103e-01
  1.01908795e-01 3.23612959e-03]
 [1.08429762e-02 5.19469865e+00 3.16187155e+00 ... 1.22398356e+00
  1.48468684e-01 4.65627118e-01]]
```

In above screen in first line we can see database contains 784 features in each template image and after applying PCA, selected features are 60 and then displaying encoder values after encryption with Key and this encoded features will get store in cloud instead of storing plain template so from above encoded values attacker wont gain access to server. Now click on 'Encoder Training using GMM Key' button to train GMM and get below output.

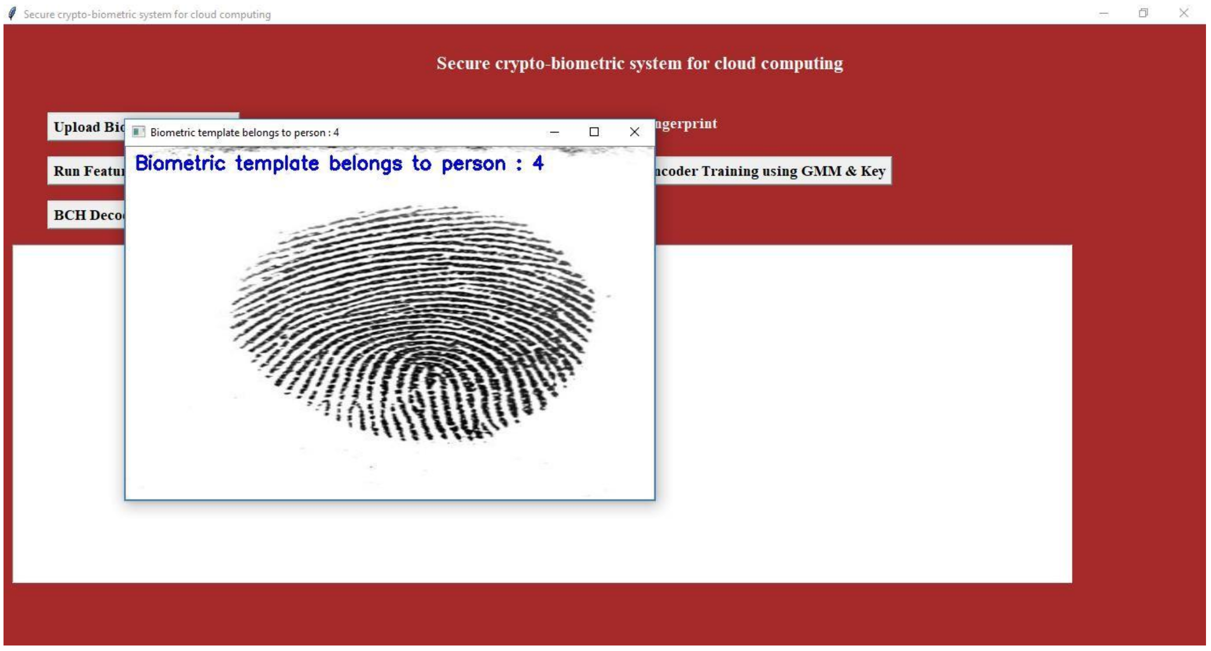

Secure crypto-biometric system for cloud computing
Secure crypto-biometric system for cloud computing
Upload Biometric Database
E:/Vithal/Aug22/CryptoBiometric/Fingerprint
Run Features Extraction
Run Features Selection & BCH Encoder
Encoder Training using GMM & Key
BCH Decoder Verification
Exit

```
Encoder training completed using GMM
```

In above screen GMM training completed and now click on 'BCH Decoder Verification 'button to upload TEST template and get below output

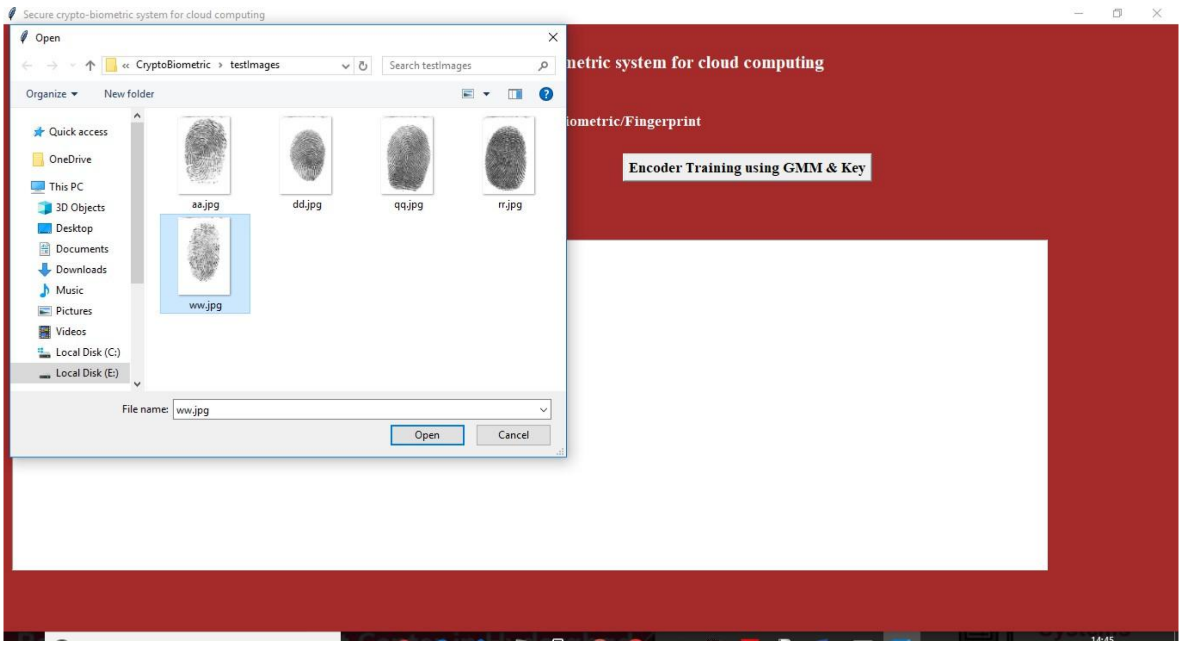


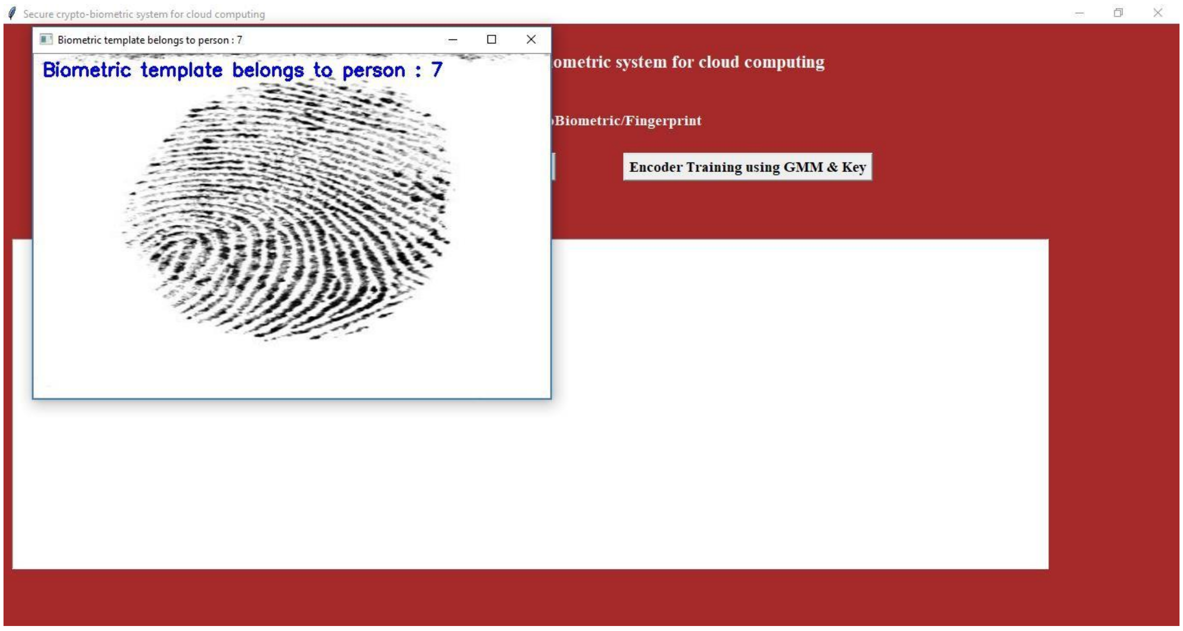
In above screen selecting and uploading 'dd.jpg' template and then click on 'Open' button to get below output.


Secure crypto-biometric system for cloud computing
Secure crypto-biometric system for cloud computing
Upload Bio
Run Featu
BCH Deco
ngerprint
ncoder Training using GMM & Key
Biometric template belongs to person : 4

Biometric template belongs to person : 4

In above screen we can see uploaded template belongs to person4 and similarly you can upload and verify other templates




Secure crypto-biometric system for cloud computing
Biometric template belongs to person : 7
ometric system for cloud computing
Biometric/Fingerprint
Encoder Training using GMM & Key

Biometric template belongs to person : 7

9. CONCLUSION AND FUTURE SCOPE

9.1 Conclusion

The development of a secure crypto-biometric system for cloud computing addresses critical concerns regarding data security in cloud environments. By employing a combination of Hidden Markov Models (HMM), Universal Background Model (UBM), and encryption techniques such as Elliptic Curve Cryptography (ECC) and Advanced Encryption Standard (AES), the proposed system ensures the confidentiality and integrity of biometric data stored in the cloud. The integration of feature extraction, selection, and encoding processes, along with GMM training, demonstrates a comprehensive approach to biometric template protection. Furthermore, the evaluation of ECC and AES encryption algorithms highlights the efficiency and effectiveness of ECC in terms of execution time. Overall, this system provides a robust solution for safeguarding sensitive biometric information in cloud-based applications.

9.2 Future Scope

There are a few aspects in our crypto biometric system that can be improved. An important limitation is the required time to process biometric data in a client application. A complex combination of models may lead to excessive time to process user biometric information (a couple of seconds in the case of a 16-UBM system as previously described), affecting response time and the interactivity of the applications that rely on our schema. A thorough investigation to improve process time should be carried out. In this line, cloud computing resources could be very useful. Accordingly, the study of a secure way to process biometric data keeping it safe and inaccessible to third parties, including the cloud provider itself, is a future line of research. Finally, a complete Software as a Service solution can be developed to provide the functionality of our approach as a service.

10.REFERENCES

[1] Balachandra Reddy Kandukuri, Ramakrishna Paturi V., Atanu Rakshit, Cloud Security Issues IEEE International Conferenceon Services Computing 2009,IEEESCC2009:517-520, Bangalore, India, Sep 2009.

[2] R. Chow, P. Golle, M. Jakobsson, E. Shi, J. Staddon, R. Masuoka, and J. Molina, Controlling data in the cloud: outsourcing computation without outsourcing control. In Proceedings of the 2009 ACM Workshop on Cloud Computing Security (CCSW '09): 85-90, ACM, New York, Nov 2009.

[3] S.Yu,C.Wang,K.Ren,and W.Lou, Achieving Secure,Scalable, and Fine-Grained Data Access Control in Cloud Computing, Proceedings of the 29th conference on Information communications, IEEE INFOCOM 2010: 534-542, San Diego, CA, USA, Mar 2010.

[4] C. Wang, Q. Wang, K. Ren, and W. Lou, Ensuring data storage security in Cloud Computing.17thInternationalWorkshoponQualityofService,IWQoS2009:1-9,Charleston, SC, USA, Jul 2009.

[5] S. Creese, P. Hopkins, S. Pearson, and Y. Shen, Data ProtectionAware Design for Cloud Services. Proceedings of the 1stInternational Conference on Cloud Computing, Cloud Com'09, LNCS 5931/2009: 119- 130, Beijing, China, Dec 2009.

[6] Q.Wang,C.Wang,J.Li,K.Ren,W.Lou,Enabling Public Verifiability and Data Dynamics for Storage Security in Cloud Computing, Computer Security ESORICS 2009, LNCS 5789, Springer: 355-370, Saint-Malo, France, Sep 2009.

[7] P. Tuyls, E. Verbitskiy, J. Goseling, and D. Denteneer, Privacy protecting biometric authentication systems: an overview, EUSIPCO 2004: XII European Signal Processing Conference: 1397-1400, Vienna, Austria, Sep 2004.

[8] D. Kesavaraja, D. Sasireka, and D. Jeyabharathi, Cloud Software as a Service with Iris Authentication,Journal of Global Research in ComputerScience,1(2):16-22 September 2010.

[9] S.Suryadevara,S.Kapoor,S.Dhatterwal,R.NaazandA.Sharma,Tongueasa Biometric Visualizes New Prospects of Cloud Computing,2011 International Conference on Information and Network Technology, IPCSIT 4(2011): 73-78, Chennai, India, Apr 2011.

[10] N.K. Ratha, J.H. Connell, and R. Bolle, Enhancing Security and Privacy of Biometric- Based Authentication Systems, IBM Systems Journal, 40(3): 614–634, 2001.

[11] A.K.Jain,K.Nanda kumar,A.Nagar, Biometric Template Security, EURASIP Journalon Advances in Sign. Proc., Special Issue on Biometrics: 113:1–113:17, 2008.

[12] A.Goh,D.C.L.Ngo, Computation of Cryptographic Keys from Face Biometrics,7th IFIP- TC6 TC11 International Conference, CMS 2003, Lecture Notes in Computer Science, 2828/2003: 1-13, Torino, Italy, Oct 2003.

[13] N. Ratha, S. Chikkerur J. H. Connell, R. M. Bolle, Generating Cancelable Fingerprint Templates, IEEE Transactions on Pattern Analysis and Machine Intelligence, 29(4):561-572, 2007.

[14] E. Maiorana, P. Campisi, J. Fierrez, J. Ortega-Garcia, A. Neri, Cancelable templates for sequence-based biometrics with application to online signature recognition,IEEETransactions on Systems, Man and Cybernetics Part A, 40(3): 525-538, 2010.

[15] D.A. Reynolds, T.F. Quatieri, R.B. Dunn, Speaker verification using adapted gaussian mixture models, Digital Signal Processing, 10(1-3): 19-41, 2000.

[16] P.Kenny,G.Boulianne,P.Dumouchel,Eigen voice Modeling With Sparse Training Data, IEEE Transactions on Speech and Audio Processing, 13(3): 345-354, 2005.

[17] C. Vielhauer, R. Steinmetz, Handwriting: Feature Correlation Analysis for Biometric Hashes, EURASIP Journal on Applied Signal Processing, 2004(4): 542-558, 2004.

[18] A.Juels,M.Wattenberg,A Fuzzy Commitment Scheme, CCS99 Sixth ACM Conference on Computer and Communication Security: 28- 36 Singapore, India, Nov 1999.

[19] P. Tuyls, A. Akkermans, T. Kevenaar, G.J. Schrijen, A. Bazen, R. Veldhuis, Practical biometric template protection system based on reliable components, Audio- and Video-Based Biometric Person Authentication (AVBPA):436-446, Hilton RyeTown, NY, USA, Jul2005.

M. Van der Veen, T. Kevenaar, G.-J. Schrijen, T.H. Akkermans, and F. Zuo, Face biometrics with renewable templates, SPIE Conference on Security, Steganography, and Watermarking of Multimedia

Chapter 1

INTRODUCTION

Chapter 2

LITERATURE SURVEY

Chapter 3

METHODOLOGY

Chapter 4

OPERATING ENVIRONMENT

Chapter 5

SYSTEM DEVELOPMENT ENVIRONMENT

Chapter 6

SYSTEM TESTING

Chapter 7

RESULTS

Chapter 8

SAMPLE CODE

Chapter 9

CONCLUSION AND FUTURE SCOPE