Jämförelse mellan olika krypteringsmetoder som hjälp för att säkra innehållet i Dropbox

How to secure your Dropbox content by using different encryption methods, a comparison

Network security 60 ECTS

Halmstad 2018-10-05
Rajesh VA sudevan
Abstract
In recent years, there has been an enormous change in cloud computing technologies. The concept of cloud computing or pay as you use model has caused many businesses to move into a new dimension. There are many cloud-based applications and cloud storage services exist in the market, e.g., Dropbox. The security of the saved document in Dropbox is a significant challenge. This thesis is concentrated on the improvement of the extended security in Dropbox by adding two encryption methods. Encryption methods such as AES (Advanced Encryption Standard) and RSA (Rivest, Shamir and Adelman) are the best suitable methods for cloud-based application has been used. This thesis shows RSA is more secure over AES whereas AES performs better for larger data compared to RSA.
**List of Abbreviation**

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IT</strong></td>
<td>Information Technology</td>
</tr>
<tr>
<td><strong>IAAS</strong></td>
<td>Infrastructure as-a-Service</td>
</tr>
<tr>
<td><strong>SAAS</strong></td>
<td>Software-as-service</td>
</tr>
<tr>
<td><strong>PAAS</strong></td>
<td>Platform-as-service</td>
</tr>
<tr>
<td><strong>SAML</strong></td>
<td>Security Assertion Markup Language Standard</td>
</tr>
<tr>
<td><strong>SOAP</strong></td>
<td>Simple Object Access Protocol</td>
</tr>
<tr>
<td><strong>OPE</strong></td>
<td>Order Preserving Encryption</td>
</tr>
<tr>
<td><strong>FPE</strong></td>
<td>Format Preserving Encryption</td>
</tr>
<tr>
<td><strong>PDP</strong></td>
<td>Provable data possession</td>
</tr>
<tr>
<td><strong>SLA</strong></td>
<td>Service Level Agreement</td>
</tr>
<tr>
<td><strong>E-RSA</strong></td>
<td>Efficient RSA</td>
</tr>
<tr>
<td><strong>EAMRSA</strong></td>
<td>Encrypt Assistant Multi-Prime RSA</td>
</tr>
<tr>
<td><strong>RSA</strong></td>
<td>RIVEST-SHAMIR-ADLEMAN</td>
</tr>
<tr>
<td><strong>KIST</strong></td>
<td>Splay tree encryption</td>
</tr>
<tr>
<td><strong>HASBE</strong></td>
<td>Hierarchical attribute-set-based Encryption</td>
</tr>
<tr>
<td><strong>AES</strong></td>
<td>Advanced Encryption Standard</td>
</tr>
<tr>
<td><strong>SSL</strong></td>
<td>Secure Sockets Layer</td>
</tr>
<tr>
<td><strong>TLS</strong></td>
<td>Transport Layer Security</td>
</tr>
<tr>
<td><strong>API</strong></td>
<td>Application program interface</td>
</tr>
</tbody>
</table>
# Table of contents

## Contents

Abstract ........................................................................................................................................... 1  
List of Abbreviation  italic ............................................................................................................. 2  
Table of contents .......................................................................................................................... 3

### Chapter 1:

1 Introduction .................................................................................................................................... 5
   1.1 Research Problem .................................................................................................................... 5
   1.2 Motivation ............................................................................................................................... 6
   1.3 Contribution ............................................................................................................................ 6

### Chapter 2:

2 Background / Related Work ........................................................................................................ 7
   2.1 Cloud Computing and Services ............................................................................................. 8
   2.2 Cloud Models .......................................................................................................................... 10
   2.3 Data Security in Cloud Computing ......................................................................................... 11
   2.4 LITERATURE SURVEY ......................................................................................................... 12
      2.4.1 Data Protection .................................................................................................................. 14
   2.5 Encryption .............................................................................................................................. 15
      2.5.1 CLOUD SUITABLE ENCRYPTION METHOD ................................................................. 17
      2.5.2 Tokenization ...................................................................................................................... 18

### Chapter 3:

3 Dropbox in Cloud Computing ...................................................................................................... 18
   3.1 Sharing in Dropbox .................................................................................................................. 19
   3.2 Dropbox Security weakness ................................................................................................... 20

### Chapter 4:

4 Implementation ............................................................................................................................. 21
   4.1 Development Requirements ................................................................................................... 21
      4.1.1 Eclipse ............................................................................................................................. 22
      4.1.2 Wampserver .................................................................................................................... 22
      4.1.3 JAVA .............................................................................................................................. 23
   4.2 API creation in Dropbox .......................................................................................................... 24
      4.2.1 Platform Independent ..................................................................................................... 24
   4.3 End user Application .............................................................................................................. 28
      4.3.1 Authorization .................................................................................................................... 28
Chapter 1:-

1 Introduction

There are many cloud storage services which allow the users to access the documents at any time and from anywhere. These services are also platform independent and could be accessed through any devices such as PC(Personal Computer), Tablet or Smartphone. This easy access to these storage services made people adopt it and share their data without knowing where their information is stored. There are cloud storage services such as Dropbox, Google Drive, Microsoft SkyDrive, etc. These services are used by many people, and they don't know actually where and who are their service providers.

Though different services provide different security to data that are stored in their storages, it might be not satisfied for all the customers who use it. There many users who usually share their data through the mail couldn’t do the same for these cloud storage services because of the security concerns. Dropbox is one among the services which offer vast security but still the security of the data in transit should be considered for safer use of personal documents.

Encryption in cloud storage services is the major way to improve the security. This thesis has given an extended security method to Dropbox which is one of the well-known cloud storage services by providing encryption through a proposed end user application. This application is used for the secure transformation of data transferred between user and Dropbox. The encrypted way of sending and receiving data is the safest way of transferring information in cloud computing.

1.1 Research Problem

This study aims to illuminate the effects of extended security method in Dropbox. When the security issues are prioritized the major problem that troubles the customers likely to move into cloud environment is Data security.

After making a detailed study on the above aspects, lack of security in Data protection and Data transparency is considered to be the major defect. One potential solution to this issue will be protecting the data which is at transit and rest. This study is motivated by two research questions:


2. Which encryption method has more advantage than another in cloud storage services?
1.2 Motivation

This study is motivated to improve Dropbox security by improving the security in data transit. The main objective is to concentrate on the data protection layer and its properties to improve the security in the cloud. By once implementing the additional security in Dropbox’s data protection and implementing it. It is possible to extend the method to secure the data in other cloud services.

1.3 Contribution

This thesis aims improving security in cloud storage services and the security of the data which is in transit. To do that, a literature survey has been made on different encryption methods; then a suitable method has been selected to be implemented. This thesis has concentrated on two areas for the improvement of security in Dropbox:

1. On the data in transit and data at rest to improve the security.
2. On providing additional security in the data transit through encryption method.

An end user application is created in such a way that Dropbox is connected to the external end user application as dictated in the dropbox developer’s page. After creating the application, the text files are sent to Dropbox using two encryption methods to ensure that the data is secured when it is in transit and at rest.
Chapter 2: -

2 Background / Related Work

In cloud computing the major problem is with the storage of data. The customers are not reliable with the provider’s security policy. A basic data model consists of the following:

Data model:

In cloud computing security, the major concern is providing security in Data protection. Usually the data security could be defined under three categories:

1. **Authentication**
2. **Encryption and Protection**
3. **Recovery**

The major existing Data model issues in IaaS challenges are:

- **Identity**: Is that the precise user who is accessing my contents and data.
- **Data Location**: Do I know where the data and storage is located geographically.
- **Availability**: The availability of the data to users irrespective of time.
- **Notification**: When there is any intrusion on data by any unauthorized user or by some malicious threat, do I get immediate notification [8].

Here the security and protection of data in Hybrid model could be improved by improving the three categories mentioned above. Usually a secured data is confirmed under first two categories of either authorization or through Encryption and protection. The authorization framework does not extended to cloud since the organization cannot merge cloud security data with its own security metrics and policies but the growing number of cloud providers support g Security Assertion Markup Language standard, it is used by the administer users and authenticate them before providing access to application and data these SAML requests messages are mapped over the Simple Object Access Protocol (SOAP). The other way to improve security in cloud is to concentrate
on the encryption and protection. Many existing cloud providers like VMware, Cloud Stack, or others started adopting this as the best method to overcome security issues.

Protecting the data in cloud and providing transparency of the data when migrating to cloud is important. This study intended to describe that there are various problems associated with the concern study on Data security. As for Data security the biggest problem faced by the providers is the security policy they provide and for customers the privacy of their data at rest. Since many customers share the same underlying infrastructure and there are many possibilities for the data loss or data steal. Considering the facts the data protection plays a major role in the cloud computing security. In the recent years online storage companies has acquired an increasing commercial success. Many applications like iCloud, Dropbox, etc. are well known to billions of people for its success. In this case the service providers buy or rent a huge number of servers and by exploiting such a large number of servers, information protection can be achieved through suitable data encoding and slicing. In this case, the client or provider software decomposes the file into smaller sub-files which are properly processed and sent to the storage sites [9]. When a client needs to retrieve specific part of a file be a sub-file which he has to retrieve it from the original set of files then he has to spend a huge initial cost for the need of high performances. To overcome this problem they have introduced a new emerging approach using cooperative storage. These storages are capable of retrieving information in nodes and could deliver in ease.

Also the other issue associated with data protection is data transparency the customers need to know what the data does when it is at rest. This transparency enables to know whether there is any leakage of data or any intrusion in the database. The data which is in transit may also leak if there is no proper passage i.e. hijacking of data is possible, to overcome all these issue and to improve the data security many researchers concentrate on the authentication and authorization of the data before entering into the cloud since the data are managed by an un-trusted server.

The existing technology and study states that there are many authentication techniques being suggested and some of the famous one are to authorize through fingerprint and even QR barcode [9]. Though many developing techniques giving hands to improve the security threats the basic encryption is needed and being used to transfer data safely. In the current environment the data at the third party server is encrypted and can be fetched by an encrypted query or decrypted at the receivers end. Also existing problems state that some of the database encryption algorithm does not support wide range of queries, also the encryption is defined as the two layered system where double layered and single layered encryption is used for sensitive and non-sensitive data respectively. The algorithm used for this kind of encryptions is Order Preserving Encryption (OPE) and Format Preserving Encryption (FPE) [11]. But still encryption method of securing data is risky from the view of data hijackers, if the hackers crack the key then it easy to read and use the data which is sent and received.

2.1 Cloud Computing and Services
Cloud computing is a specialized distributed paradigm which has become an alternative for using local servers to handle applications. It is a new trend of computing where readily available computing resources are exposed as services. Hardware and the system software in the data centers provide these services. Cloud computing tremendously getting its growth in recent years, even the small industries has found the advantages of cloud computing and start to deploy their data and applications in the
cloud environment. Cloud computing is the sharing of data processing tasks from the large group of remote servers and to make use of its service and resources. In other words we can say that it is a utilization computing that can avail the computing power or resources on rent.

From the birth of IT (Information Technology) Industry has seen many changes and people started adopting all the changes. The major development and improvement in cloud has taken place in recent years, but in the history of IT industry “Cloud Computing” is considered to be one of the big revolutionary change which brought both relief and confusion from the business view. Many customers have to buy and pay for all the equipment. This made them to spend more money. The major requirement of cloud is to reduce the cost on spending for each and every product. This pay as you use model made them happier and doubtful how it will work out.

For example: considering an IT company it is difficult for the company to provide different software and operating system to the employees since they have to pay separately for each and everything, so here what they will do is just buy and install one product and give access to its employees to use it. Here the software is bought and operated in the cloud’s centralized data storage. Considering the advantage it is not easy to shift from the existing set-up to cloud for the customers because security is considered to be the big issue.[1]

Cloud Computing Security as “Cloud computing security (sometimes referred to simply as "cloud security") is an evolving sub-domain of computer security, network security, and, more broadly, information security. It refers to a broad set of policies, technologies, and controls Deployed to protect data, applications, and the associated infrastructure of cloud computing.”[15]

In this thesis securing the IAAS (Infrastructure as-a-Service) is most important because the security requirements for using Infrastructure as-a-service (IAAS) considered as same as your own data center. This area of operation is essentially refers to purchasing the basic storage, processing power and networking to support the delivery of cloud computing applications. This clearly shows that this area of operation has more control over the IT infrastructure in any organization that meets with the above requirements. But the cost structure is the consideration factor. The security issues in IaaS are data leakage protection and usage monitoring, Authentication and Authorization, End to End encryption[2].

The three building blocks of cloud are considered as the following:
(SAAS) Software-as-service, (PAAS) Platform-as-service, (IAAS) Infrastructure-as-service
All these services allow running application and running data online, however each offers different levels of user flexibility and control.

SAAS
Software-as-service: This allows running existing online application. This is considered as the front end of the model where all the applications will reside here. It is actually accessible from any computer and also facilitates collaborative work.

PAAS
Platform-as-service: Allow users to create their own cloud applications and using supplier specific languages and tools i.e. it provides environment and tools for creating new online applications. This allows the users to create more applications very rapidly at low cost.

IAAS
Infrastructure-as-service: Allow users to run any applications they please on cloud hardware of their own choice.

2.2 Cloud Models

The cloud is deployed into three models they are: Public, Private, and Hybrid.

Public
This model is operated by the third party cloud provider and in this model the resources and services are shared by multiple customers (multi-tenancy) where its infrastructure is located at Off-premise.[3] This model comprises with the following security issues like unauthorized access of data(Multi-tenancy risks), lack of transparency(control and visibility), data retention, compliance requirements.

Private
A well developed and designed private cloud would be prepared for an evolution to Hybrid cloud. This model is operated by both organization and the cloud provider and where the resources and services are shared within the organization (single tenant) where there are two variants in this model, on premise private clouds and externally hosted private clouds [6]. In regards to security here the organization knows who has the access to firewalls and resources yet the major threats are Security Responsibility, Share tenant model, (Hypervisor vulnerability) Virtualization, security attack targets[6]. In this report I have proposed the way to improve security in Hybrid cloud environment which is accessible through IAAS.

The well-known approach to cloud computing and cloud Storage services from the above deployment model are Hybrid cloud. This allows data to be held offsite with the cloud provider, but also allows for data to be held internally within the organization. Companies of all types use a combination of private and public cloud services operating together in a hybrid environment. Future IT industry is dependent on such Hybrid environment. Since Hybrid model is the most dependent model by the cloud providers and customers. I was motivated to make a research on the Hybrid cloud’s security issues and the ways to improve security in it.

The security in Hybrid cloud, the security concerns are totally based on industry, services delivered, compliance and other auditing requirements and many other concerns are based on non-cloud approach. Some of the other issues are fairly important for any organization to adopt cloud like security of data when migrating to cloud and protection of data in cloud, also the recovery methods to retrieve the lost or migrated data. Securing data is the major problem in Cloud Computing [5].

Hybrid
This cloud owns the feature of more than one cloud types whereas this offers the advantage of both public and private cloud features, some of the benefits of hybrid
cloud over other is this is cost effective, more agility, and easy maintenance (Access through API's)[5]. Some of the security issues faced in Hybrid cloud is Confidentiality and integrity, Scalability, Data Segregation, Network Security and Security Policies.

The next-generation hybrid cloud environment offers flexibility and cost control which are in great demand. For a consistent approach towards hybrid cloud, standardization should be done.

The most common way organizations create hybrid environments is through integration at the data level. Integrating information across multiple public- and private-based data sources might be challenging, though all the data’s are highly secured in the cloud’s datacenter when we try to get on the data from the public cloud sources, the complexity increases eventually.

Hybrid cloud provides a clear use case for public cloud computing. Existing IT infrastructure especially (storage) occur in public cloud environments rest in on-premise. Hybrid cloud helps us all get better at understanding what compute cycles and data have to be kept local and what can be process remotely.

Also when it comes to third party data processing there are many laws with extended data protection. Also for the EU, the support of third parties for data processing is permitted [12].

In third party Data control there are additional security issues such as due diligence, Transitive nature of contracts, Auditability, Contractual obligations and Cloud provider espionage.

2.3 Data Security in Cloud Computing

The security for Data in cloud computing should be concentrated on the following aspects:

Data in transit

From the name Data in transit it is clear that any data which is in travel or transit is considered. In the perspective of security is concerned the data which is in transit has higher probability for being theft or hacked, i.e. When two nodes in a network are communicated, the data transferred in-between the nodes are considered as the data in transit. Protecting the data in transit is one of the major security lacks in cloud. Some information could be downloaded or emailed and the time which it travels is in risk. When data is in transit, it can be intervened or meddled at any point of the travel. Data encryption in transit Standard web traffic is not encrypted. This leads to the risk in stealing or loss of data. There are better chances to protect the data by sending it in an encrypted form.
**Data at Rest**

The data storage in the data center is also called as the data in rest. This is the secured part of the database and also needs more security so that no intrusioin takes place from the external or unauthorized source.

**Processing of data including Multi tenancy:**

When it comes to cloud it is a multitenant environment. The sources in the cloud are share among the other tenants in the cloud. So considering the data, processing of it should make available to all the tenants who are in need of it. In other words it can be said that the availability of data should be considered.

**Data Lineage**

Data lineage is to find what happens to data when it moves around in a certain amount of time i.e. To see data from its source of origin till the destination it reaches.

**Data Provenance**

This similar to that of data lineage but also helps to trace the data in which database it actually resides in reference [7]

Out of all the above data security issues data in transit and data at rest are considered in this thesis as the major issues and how it can be secured.

### 2.4 LITERATURE SURVEY

From this paper they introduce a model for provable data possession (PDP) that allows a client that has stored data at an untrusted server to verify that the server possesses the original data without retrieving it. The model generates probabilistic proofs of possession by sampling random sets of blocks from the server, which drastically reduces I/O costs. The client maintains a constant amount of metadata to verify the proof. The challenge/response protocol transmits a small, constant amount of data, which minimizes network communication. Thus, the PDP model for remote data checking supports large data sets in widely-distributed storage systems. The two provably-secure PDP schemes that is more efficient than previous solutions, even when compared with schemes that achieve weaker guarantees. In particular, the overhead at the server is low (or even constant), as opposed to linear in the size of the data. Experiments using our implementation verify the practicality of PDP and reveal that the performance of PDP is bounded by disk I/O and not by cryptographic computation [23].

Also in Privacy-Preserving Audit and Extraction of Digital Contents there is a growing number of online services, such as Google, Yahoo!, and Amazon, are starting to charge users for their storage. Customers often use these services to store valuable data such as email, family photos and videos, and disk backups. Today, a
customer must entirely trust such external services to maintain the integrity of hosted data and return it intact. Unfortunately, no service is infallible. To make storage services accountable for data loss, we present protocols that allow a third party auditor to periodically verify the data stored by a service and assist in returning the data intact to the customer. Most importantly, our protocols are privacy-preserving, in that they never reveal the data contents to the auditor. Our solution removes the burden of verification from the customer, alleviates both the customer’s and storage service’s fear of data leakage, and provides a method for independent arbitration of data retention contracts [24].

Cloud computing is the long dreamed vision of computing as a utility, where data owners can remotely store their data in the cloud to enjoy on-demand high-quality applications and services from a shared pool of configurable computing resources. While data outsourcing relieves the owners of the burden of local data storage and maintenance, it also eliminates their physical control of storage dependability and security, which traditionally has been expected by both enterprises and individuals with high service-level requirements. In order to facilitate API deployment of cloud data storage service and regain security assurances with outsourced data dependability, efficient methods that enable on-demand data correctness verification on behalf of cloud data owners have to be designed. In this article we propose that publicly auditable cloud data storage is able to help this nascent cloud economy become fully established. With public auditability, a trusted entity with expertise and capabilities data owners do not possess can be delegated as an external audit party to assess the risk of outsourced data when needed. Such an auditing service not only helps save data owners’ computation resources but also provides a transparent yet cost-effective method for data owners to gain trust in the cloud. We describe approaches and system requirements that should be brought into consideration, and outline challenges that need to be resolved for such a publicly auditable secure cloud storage service to become a reality [25].

The Storage outsourcing is a rising trend which prompts a number of interesting security issues, many of which have been extensively investigated in the past. However, Provable Data Possession (PDP) is a topic that has only recently appeared in the research literature. The main issue is how to frequently, efficiently and securely verify that a storage server is faithfully storing its client’s (potentially very large) outsourced data. The storage server is assumed to be untrusted in terms of both security and reliability. (In other words, it might maliciously or accidentally erase hosted data; it might also relegate it to slow or off-line storage.) The problem is exacerbated by the client being a small computing device with limited resources. Prior work has addressed this problem using either public key cryptography or requiring the client to outsource its data in encrypted form [26].

The Provable data possession is a technique for ensuring the integrity of data in outsourcing storage service. In this paper, author proposed a cooperative provable data possession scheme in hybrid clouds to support scalability of service and data migration, in which the existence of multiple cloud service providers to cooperatively store and maintain the clients’ data is considered. Our experiments show that the verification of our scheme requires a small, constant amount of overhead, which minimizes communication complexity [27].
2.4.1 Data Protection

Regarding data protection, cloud computing raises a number of interesting issues. Data protection law is based on the premise that it is always clear where personal data is located, by whom it is processed and who is responsible for data processing. Also under the protection law the customer can choose at what locality the intended Data Could be stored also when the data is in transit it’s difficult to say where the data is exactly located.

Usually the users need the control of their data, but also require the benefits service from the application developers which they provide with that data this tension leads to the requirement of data protection. So far much standardization has been made for the user data which are at rest and in transit. Always providing a single data protection technique to solve all the issues is not possible instead it can be done by the applications and its uses, depending on the usage of the application different data protection methods can be used.

For giving a perfect data protection solution based on the application we have to follow the certain goals mentioned below:

Integrity

The user’s stored data should be able to access only through the authorized user. Data integrity is ensuring the consistency and accuracy of the data stored in the cloud’s storage; any change in the data may question the cloud provider’s security. To ensure this the data integrity law which is also called as the Service Level Agreement (SLA) should be made transparent so that any intrusion of any third party access which could be logged in.

Privacy

Private data should be kept secured that no unauthorized entity can access it.

Access Transparency

This is the log generated to keep in track of the users who are accessing the data.

Verification

Platform dependencies are made to be easy for the users to find in what code the application is running.

Rich Computation

The platform will allow efficient, rich computations on sensitive user data[16].
To overcome these problems there is a special service proposed called (DPaaS) Data-protection-as-a-service in cloud computing. Many service providers started providing this service to protect the data, but to access that users should rely on legal requirements. The major advantage of DPaaS is that it can also provide cryptographic security, robust logging and auditing to provide accountability. This additional service from the service provider benefits smaller company and developers to increase their business.

Figure 2 illustrates example architecture for exploring the DPaaS design. Here, each server contains a trusted platform module (TPM) to provide secure and verifiable boot and dynamic root of trust. This example architecture demonstrates at a high level how it’s potentially possible to combine various technologies such as application confinement, encryption, logging, code attestation, and information flow checking to realize DPaaS.

![Figure 2 Data Protection Architecture](image)

### 2.5 Encryption.

The encryption is performed with different algorithms suitable for both cloud provider and the customer. Encryption is one of the existing methods to secure cloud’s end-to-end data. This is the famous method to secure data using any cipher algorithm. But considering only this as a security method highlights few disadvantages which are to be considered (1) in encryption it can secure only the data’s which are in rest whereas the data in transit are affected. (2) The problem whether key is with the proper person or who owns it [14].

Encryption is used to protect the data, or text using encryption algorithm. This is nothing but transforming the data into cipher text and the encrypted data is sent along with the key to decrypt the text. The key is managed separately by key management techniques.
In this paper a survey is made on different encryption techniques which could suit cloud computing. In cloud computing this type of service is provided by the service providers to encrypt and decrypt the text as per the customer’s request. In cloud computing the service provider’s encryption competence should equal the sensitivity of the data. Considering the cost, many service providers are using less cost for data encryption which in turn reduces the customers trust over cloud computing.

Also in the beginning of cloud computing era, many customers felt cloud providers could manage the encryption keys which have become apparent because of the SLA’s issued by the government.

In real time there are many encryption algorithms which are used to secure the data in cloud one such method is homomorphism encryption algorithm in the cloud computing data security. This method includes key generation algorithm, encryption algorithm, decryption algorithm and additional Evaluation algorithm. In this homomorphism encryption there are two major types additively homomorphic encryption algorithm and the multiply homomorphic encryption algorithm. Usually a homomorphic encryption is to find an encryption algorithm which can be any number of addition and multiplication algorithms in the encrypted data. This type of algorithm is to ensure the safety transmission of data between the cloud and the user [17].

The following algorithms are some of the homomorphic algorithm and its characteristics

<table>
<thead>
<tr>
<th>ALGORITHM</th>
<th>TYPE OF HOMOMORPHIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSA</td>
<td>Multiplicative</td>
</tr>
<tr>
<td>Paillier</td>
<td>Additive</td>
</tr>
<tr>
<td>El Gamal</td>
<td>Multiplicative</td>
</tr>
<tr>
<td>Goldwasser-Micali</td>
<td>Additive, but it can encrypt only a single bit</td>
</tr>
<tr>
<td>Boneh-Goh-Nissim</td>
<td>Unlimited additions but one Multiplication</td>
</tr>
<tr>
<td>Gentry</td>
<td>Fully</td>
</tr>
</tbody>
</table>

Table 3.1 Different types of Homomorphic algorithms and its functionality

In today’s business the encryption is done at the client side to bring trust on the data security for the customers, in data encryption service model the data and keys are stored in different cloud storage and cryptography process here the encryption and decryption is done at the client side where the key is requested from the key cloud server and data from the data cloud server.

This paper [18] has compared and studied the strengths and weakness of six asymmetric key encryption algorithms based on security parameters, Original RSA, RSA Small-e (RSA-based algorithm with a smaller public exponent), RSA Small-d, MREA (Modified RSA Encryption Algorithm), E-RSA(Efficient RSA), and EAMRSA(Encrypt Assistant Multi-Prime RSA) where RSA stands for (RIVEST-SHAMIR-ADLEMAN).
EXECUTION TIME (MS) ACCORDING TO RSA, RSA SMALL-E, RSA SMALL-D, MREA, EAMRSA, AND E-RSA ALGORITHMS

<table>
<thead>
<tr>
<th>User No</th>
<th>Key Size (bits)</th>
<th>RSA</th>
<th>RSA Small-e</th>
<th>RSA Small-d</th>
<th>MREA</th>
<th>E-RSA</th>
<th>EAMRSA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>512</td>
<td>1145</td>
<td>863</td>
<td>1165</td>
<td>11007</td>
<td>1277</td>
<td>21558</td>
</tr>
<tr>
<td>2</td>
<td>1024</td>
<td>2999</td>
<td>2285</td>
<td>2802</td>
<td>19533</td>
<td>3160</td>
<td>30693</td>
</tr>
<tr>
<td>3</td>
<td>2048</td>
<td>4058</td>
<td>3310</td>
<td>3857</td>
<td>27275</td>
<td>4507</td>
<td>40997</td>
</tr>
<tr>
<td>4</td>
<td>3072</td>
<td>6464</td>
<td>4653</td>
<td>5202</td>
<td>33624</td>
<td>7535</td>
<td>49835</td>
</tr>
</tbody>
</table>

Table 3.2: EFFECT OF CHANGING THE KEY SIZES FROM 512 (BITS) TO 3072 (BITS) ON TOTAL EXECUTION TIME (MS) ACCORDING TO RSA, RSA SMALL-E, RSA SMALL-D, MREA, EAMRSA, AND E-RSA ALGORITHMS

From the analysis it is identified that E-RSA and RSA are best Algorithms for encrypting data in the client side, whereas E-RSA is more secure against attacks. In another study [19] eight modern encryption techniques RC4, RC6, MARS, AES, DES, 3DES, Two-Fish, and Blowfish are tested by using NIST statistical testing in the cloud computing environment on two different platforms: desktop computer and Amazon EC2 Micro Instance cloud computing environment. From the statistical analysis of the eight modern encryption algorithm the RC6, AES results are better when compared to the other algorithm. Finally when considering the cloud environment AES encryption shows better results for long run.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rejection rate</td>
<td>AES</td>
<td>DES</td>
<td>Blowfish</td>
<td>RC4</td>
<td>Twofish</td>
<td>RC6</td>
<td>3DES</td>
<td>MARS</td>
</tr>
<tr>
<td>P-Value</td>
<td>AES</td>
<td>RC6</td>
<td>3DES</td>
<td>MAR S</td>
<td>DES</td>
<td>Blowfish</td>
<td>RC4</td>
<td>Twofish</td>
</tr>
<tr>
<td>Enc/Dec Speed</td>
<td>Blowfish</td>
<td>AES</td>
<td>RC4</td>
<td>DES</td>
<td>RC6</td>
<td>MARS</td>
<td>3DES</td>
<td>Twofish</td>
</tr>
</tbody>
</table>

Table 3.3: Different algorithms which have been tested under different parameters like Rejection rate, P-Value, Encryption/Decryption speed.

From the above references and analysis the four algorithms RSA, RSA-e, RC6, and AES are proved to be efficient encryption methods.

2.5.1 CLOUD SUITABLE ENCRYPTION METHOD

When it comes to cloud suitable encryption methods the homomorphic algorithms are considered to be one of the best suitable algorithms and when the normal encryption methods are compared from the view of security defines RSA, RSA-e, RC6, AES is considered to be the best among the other encryption method. Here RSA falls under both the category and considering this method defines it to be one of the suitable methods of encryption for Cloud environment. New encryption methods such as bio-
metric encryption are used to store biometric information of the encrypted details. This algorithm uses three models Key Release, Key Binding and Key Generation [20]. Where the authentication is given more priority for the sake of security and the key is generated as per the user’s biometric information. This enhance the bio-metric confidentiality in cloud computing. Usually biometric identification includes iris, voice, fingerprint, face reorganization and etc. Biometric Encryption provides extra layer security for privacy and also for the attacker in the Cloud.

Many other new encryption methods such as key insertion and splay tree encryption (KIST), Order preserving encryption (OPE), hierarchical attribute-set-based encryption (HASBE) are proposed for the security in cloud computing. From the above theoretical study it is believed that RSA and AES are the two methods which are considered to be used as a standard encryption method. But other study proves that in the field of cloud computing, the development and usage of data is going to be tremendous and sure it requires a better security model to secure the loss of data.

By once implementing the additional security for data in Hybrid cloud we can easily secure the data present in the IaaS and we can introduce it to Public and Private Cloud in near future. To improve the security in hybrid cloud first the suitable encryption method should be selected. From the survey we consider RSA and AES encryption methods which suits the cloud computing.

2.5.2 Tokenization.

Tokenization is a process by which a sensitive data is replaced by an alternate value called token. This tokenization is widely used in the banking Industry for coding and decoding the one’s account number and other personal details. The advantages of this tokenization is that when a hacker hacks the data the hacked token value represents a dummy(null) value which as no relation with the original data. The advantages of tokenization over encryption are the value generated here cannot be reversed to get back its original data instead it should match the generated token value with the original value in the “secured” database inside a company's firewall [14]
Chapter 3

3 Dropbox in Cloud Computing

Dropbox falls under the SAAS service of the cloud computing. Dropbox is one of the top network storage services among its other competitor. Dropbox can be easily accessed and it is one of the famous cloud based application which is used to share files and other documents from both computer and smartphone. Each user should register separately to obtain certain storage space. Though dropbox is used by more no of population around the world no personal or business documents are shared because of the less security measures. Dropbox has its own problem and weakness in sharing information.

3.1 Sharing in Dropbox

Information or documents could be shared in dropbox by two ways

**Public sharing**
In this the files to be shared can be placed in a folder by the name “Public” and the URL of the corresponding folder can be shared with anyone to view the specific file that has been shared. Anyone can share the URL to share the specific file. Also there is a secret URL generated by the dropbox is an alphanumeric sequence which is used to identify the specific file. This URL is unique for every single file. However there is no authentication for the user who views the file which has been shared. So once the URL is shared between the wrong hands then the information is leaked.

**Private sharing**
This type of sharing is provided only for the folders and not for any single file separately. To use this sharing option everyone who needs to access the folder should have a separate account in dropbox. If the user is registered with the e-mail then the invitation which is sent will be initiated to the registered email address or the link is generated to share between the users in any email address [30].
3.2 Dropbox Security weakness

Nondead URL: - The URL sharing can be used to share the files between the users. When a URL which has been linked to a path has been shared with the other user, they can open the document using the link shared, also any user can open the document using the link which has been shared. Even if the creator deletes the file after sharing the document and creates a new unshared document under the same name, the other users with whom the file has been previously shared could open the document with the same URL. This is considered to be the one of the big security weakness in Dropbox.

NonHTTPS shortened URL: The user can also request the sharing URL using any smartphone which in turn returns a shortened URL in a case-sensitive alphanumeric sequence. This shortened URL actually redirects to the original URL of the shared document. This URL could be easily shared and it is not SSL protected. An eavesdropper can acquire and access the file. Moreover, if we search “Site: db.tt” on Google, 60,000 files in the alphanumeric sequence is displayed. An attacker or hacker can have a brief search to get the significant information and might get the URLs which could have been linked to the private data.

Unauthorized Sharing: User can share the file using private sharing and can also change the setting to ensure that the other user cannot invite others to view the data. However, the second user with whom the file is shared can still share the data using secret URL sharing without the knowledge of the original user. This is also a major security threat in Dropbox.

No Privacy: The identities of all the people with whom the File is shared could be seen. Also in real time anyone can share it to the third person using secret URL which is still a threat to any business. The other threat is when the user shares the URL with the new person and If that person doesn’t have dropbox account he can then create a dropbox account using any email address, so new identities can exist if the shared person uses other than his old email address. This brings the confusion with whom really the data is shared.

These problems together can act as the major weakness to the dropbox [30].
Chapter 4: Implementation

The implementation part is to show which encryption method chosen to be the best suitable for cloud. Here we chose dropbox as example as it is one of the renowned cloud storage services. In this implementation we have took two encryption methods namely AES and RSA and tried to prove which one stands better over another based on the usage efficiency time. So the file information would be hidden from the rest of users, even they possess the file's key and owner info

Presently any user with private key, file name and uploader name can access the file of the uploader.

In future, we can restrict this viewability confined to users.

For that, the uploader can share the user details where the user can access during the file upload or when the uploader try to view the uploaded file.

Here the flow of the uploaded data starts from the client side and the data is encrypted using the encryption protocol. The client’s encrypted data will initiate the pseudo timing system where the time is calculated and once the data reaches the storage then the pseudo timing stops and the pop up shows the amount of time for the information to get stored in the dropbox.

This system has its own disadvantages, as the private key sharing is one of the biggest and known problems.

As of here we have to use a well-known method of sharing the key via other resources such as mail, but in future we can combine the mail system and the dropbox user system which are confined with the same mail address and can send the encryption information where the application itself triggers the key to the mail address.

4.1 Development

4.2 Requirements

Step 1: Java language is used for the development of the end user application.

Step 2: Eclipse software is enabled to write java program.
Step 3: WampServer connectivity is verified as it is used to maintain the database. 
Step 4: Knowledge on Dropbox is required and how API is created in dropbox [21]. 
Step 5: Connectivity between the End user application and Dropbox is ensured. 

4.2.1 Eclipse

Eclipse is mainly designed for developing integrated development environments (IDEs) and arbitrary tools

Through the Eclipse Platform the following requirements are satisfied.

- Construction of a variety of tools for Building application development is supported.
- Tools to manipulate arbitrary types (e.g., HTML, Java, C, JSP, EJB, XML, and GIF) are supported.
- Facilitate seamless integration of tools within and across different content types and tool providers.
- Supports GUI and non-GUI-based application development environments.
- Platform independent i.e. (Supports Windows, Linux, Mac OS X, Solaris AIX, and HP-UX.)
- Major role of Eclipse is to provide tools with mechanisms that take to seamlessly-integrated tools. Java programming language is also written using such writing tools from eclipse. API interfaces, methods and classes are written using this mechanism [29].

4.2.2 Wampserver

The Wampserver is specially web development environment made for windows to create web applications where it mainly stands for with PHP, Apache, and a MySQL. PhpMyAdmin allows managing easily your databases.

After downloading the Wampserver it should be installed in the computer. We can activate the Wampserver by clicking the icon which is created on the desktop. After the activation of the Wampserver Type “localhost” in the browser, open Database Page Click PhpMyAdmin or Type at browser “localhost/PhpMyAdmin”. New projects are added there.

The major features of Wampserver are

- Managing the Apache and MySQL services
- Can switch between PHP, Apache and MySQL
- Access the logs and server settings
- Can define to whom the access is given (everyone or localhost) [22].

I have used wamp server to maintain my data’s such as user name, password, User’s connection details to the dropbox, etc.
4.2.3 JAVA

This chapter is about the software language and the tools used in the development of the project. The platform used here is JAVA. The Primary languages are JAVA, J2EE and J2ME. In this project J2EE is chosen for implementation.

Features of JAVA

THE JAVA FRAMEWORK

Java is a programming language originally developed by James Gosling at Microsystems and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++ but has a simpler object model and fewer low-level facilities. Java applications are typically compiled to byte code that can run on any Java Virtual Machine (JVM) regardless of computer architecture. Java is general-purpose, concurrent, class-based, and object-oriented, and is specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere". Java is considered by many as one of the most influential programming languages of the 20th century, and is widely used from application software to web applications the java framework is a new platform independent that simplifies application development internet. Java technology's versatility, efficiency, platform portability, and security make it the ideal technology for network computing. Java is everywhere from laptops to datacenters, game consoles to scientific supercomputers, smart phones to the Internet.

Objectives of JAVA

To see places of Java in Action in our daily life, explore java.com.

Advantages of using JAVA

Java has been tested, refined, extended, and proven by a dedicated community. And numbering more than 6.5 million developers, it's the largest and most active on the planet. With its versatility, efficiency, and portability, Java has become invaluable to developers by enabling them to:

- Write software on one platform and run it on virtually any other platform
- Create programs to run within a Web browser and Web services
- Develop server-side applications for online forums, stores, polls, HTML forms processing, and more
- Combine applications or services using the Java language to create highly customized applications or services
- Write powerful and efficient applications for mobile phones, remote processors, low-cost consumer products, and practically any other device with a digital heartbeat
4.3 API creation in Dropbox

API is actually meant by Application Program Interface. Dropbox has a separate information for the advanced users stating the below. The security of the stored data in dropbox is generally stored in the encrypted format, and the encryption format used when the data is at rest is AES. Also it has been designed with multiple layers of protection, including secure data transfer, encryption network configuration and application.

There is a special option in dropbox for the third party application using dropbox integration, to access this we need to ask permission. Also the access is granted by authorizing the third party app from the individual’s dropbox account.

The following shows how the dropbox has given information for the advanced users.

For advanced users

“As described in dropbox website

- Dropbox files at rest are encrypted using 256-bit Advanced Encryption Standard (AES).
- Dropbox uses Secure Sockets Layer (SSL)/Transport Layer Security (TLS) to protect data in transit between Dropbox apps and our servers; it's designed to create a secure tunnel protected by 128-bit or higher Advanced Encryption Standard (AES) encryption.
- Dropbox applications and infrastructure are regularly tested for security vulnerabilities and hardened to enhance security and protect against attacks.
- Two-step verification is available for an extra layer of security at login. You can choose to receive security codes by text message or via any Time-Based One-Time Password (TOTP) apps, such as those Listed here

- Public files are only viewable by people who have a link to the file(s) [21].”

New API is available for dropbox in the developer’s page
https://www.dropbox.com/developers there is an option for creating application integrating dropbox in it [21].

From the developer’s page instruction we can create a Platform independent API.
Platform independent is meant by any software or application could run on any hardware platform or any operating system irrespective of the language in which the application is developed. dropbox can support Swift, Python, .NET, Java, JavaScript, PHP, Ruby, HTTP [21].

4.3.1 Platform Independent

For this, the application console in the developer home menu is considered; by logging in we can find the options to create new application as shown in Figure 5.1.

This Dropbox API for mobile and web applications offered by the dropbox has already built in security and the advanced security of the external application created will add security to the data even in the transit.

More supported platforms

Version 0 of the API was designed for mobile applications; better support for web applications has been added! This includes new tutorials, documentation, and SDKs for Python, Ruby, and Java developers.
The application is created in the developer home and the app generates the app key in the application and the app key is submitted at the end user application which helps to sync the dropbox. Once the API (Application program interface) is created it will be listed in the tab your apps. This is the API created in the dropbox side and it can be connected with the end user app using the application with the end user application.
Figure 4 shows how to create a new dropbox platform application and we can choose from the option that what type of application we want to create. The Drop-ins app and Dropbox API app are two option provided by the dropbox. Also the additional options such as limitations and access can also be given when creating the app as shown above.
The figure 5 shows how the application is finally created in the dropbox. By creating the application
The above app key is used to authenticate the API by using it in the external application. This app key is stored as a product key in the application at the end user side. Now the dropbox is connected to the external application which is created. Now when the application is active the data can be transferred using the wampserver to maintain the data in it.

4.4 End user Application

This is the application which is created using programming language java. Here in this chapter the execution and the working of the end user application is explained. This application is segregated into different sections like authorization, New user creation, login page, upload file, update file and download file.

4.4.1 Authorization

For every application an authorization should be given to access the dropbox’s service. To provide this the following authorization tab is used. Here by validating the
code and saving it will make the communication between the dropbox and the application.

![Dropbox Authorization](image)

**Figure 7**

The app key and the app secret key from the dropbox are used to authorize the application. Once the app key is given an URL authorization is received in the specific column and when pasting the URL in the browser an authorization code is generated and by validating the code the application at the end user side is authorized and by saving the authorization key at the workspace will make the software work.

### 4.4.2 Create New User

When the program is initialized the specific user could Signup into the cloud account as mentioned in the **figure 8**. The Email here which is given should match the dropbox mail id so that the files updated through this software will be updated directly to the Dropbox.

Once the signup is done submit it. New user in the cloud account is created.
4.4.3 Login page

This is the main screen in the above figure 9 will run once the program is initialized.

1. Create account
   - Signup feature will be opened
2. Authorize application - To Authorize the application using app key

The **multiple Public Auditing** here is for my future work, as of now that has been kept as a non-working icon. Using the above screen the data owners will login to upload their documents.

**Figure 10**

Above **figure 10** shows upload blocks to upload the text data which needs to be uploaded, Update the data which is uploaded previously and End user
4.4.4 Upload File

So in the implementation part we used the concept of double encryption method where the dropbox itself will encrypt the data which is stored in the dropbox with AES encryption. The main aim in my thesis is to provide the data protection to the one which is even in transit. The text file is uploaded using this application shown in figure 11, the application is created in such a way that file is uploaded using your user name and using the key symbol in figure 11 the private key is generated, and using the browse option the document is uploaded as a file. When the file is uploaded you
could see the encrypted file in your dropbox account as shown below.

The rajraj.txt is the file which is uploaded using the application and the file is uploaded in the dropbox as shown in the figure 12. The figure 13 shows how the file is uploaded in the dropbox as an encrypted document. Only this document is viewed when the data is hacked even when it is in transit. The document is encrypted using RSA algorithm which is one of the homomorphic algorithm that perfectly suits cloud.
4.4.5 Update File

The uploaded file could be edited or updated as shown in the Figure 14. To use this feature the user should remember the private key which has been used while uploading the file. Once the private key is used the content in the document will be displayed in the column, after the text has displayed we can edit or delete the content and can update in the dropbox using the same private key.
4.4.6 Download File

The figure 15 is for the end user where the document can be downloaded using the same private key. The document can also be shared safely with anyone who uses the same private key. So that the sharing of file is safe and secured.
5 Experimentation Result

I have used the above stated end user application for my experimentation. The time taken to upload and download data to dropbox is calculated, the time is calculated once the uploaded data hits the dropbox server and vice versa. In this application the two way protection method is implemented [28]. The two encryption methods which are defined to be the best suitable methods are considered from different surveys. Out of all the methods AES and RSA are satisfying the required security. The dropbox platform is taken to test the end user application and both the encryption methods are used to find the best suitable.

AES

<table>
<thead>
<tr>
<th>User Data</th>
<th>File Size(KB)</th>
<th>Time Required for Uploading /sec</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>AES</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0.2718</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>0.4211</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>0.6064</td>
</tr>
<tr>
<td>4</td>
<td>15</td>
<td>0.6486</td>
</tr>
<tr>
<td>5</td>
<td>20</td>
<td>0.6223</td>
</tr>
<tr>
<td>6</td>
<td>25</td>
<td>0.5919</td>
</tr>
<tr>
<td>7</td>
<td>30</td>
<td>0.5107</td>
</tr>
<tr>
<td>8</td>
<td>35</td>
<td>0.5662</td>
</tr>
<tr>
<td>9</td>
<td>40</td>
<td>0.5401</td>
</tr>
<tr>
<td>10</td>
<td>45</td>
<td>0.5808</td>
</tr>
<tr>
<td>11</td>
<td>50</td>
<td>0.5475</td>
</tr>
<tr>
<td>12</td>
<td>55</td>
<td>0.6209</td>
</tr>
<tr>
<td>13</td>
<td>60</td>
<td>0.5741</td>
</tr>
<tr>
<td>14</td>
<td>65</td>
<td>0.5822</td>
</tr>
<tr>
<td>15</td>
<td>70</td>
<td>0.5071</td>
</tr>
<tr>
<td>16</td>
<td>75</td>
<td>0.5405</td>
</tr>
<tr>
<td>17</td>
<td>80</td>
<td>0.6248</td>
</tr>
<tr>
<td>18</td>
<td>85</td>
<td>0.5337</td>
</tr>
<tr>
<td>19</td>
<td>90</td>
<td>0.5624</td>
</tr>
<tr>
<td>20</td>
<td>95</td>
<td>0.5888</td>
</tr>
<tr>
<td>21</td>
<td>100</td>
<td>0.5770</td>
</tr>
</tbody>
</table>

Table 6.1 Time taken to upload files in (KB) using AES encryption
<table>
<thead>
<tr>
<th>User Data</th>
<th>File Size(KB)</th>
<th>Time Required for Downloading /sec</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>AES</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0.1806</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>0.1989</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>0.2614</td>
</tr>
<tr>
<td>4</td>
<td>15</td>
<td>0.3594</td>
</tr>
<tr>
<td>5</td>
<td>20</td>
<td>0.3600</td>
</tr>
<tr>
<td>6</td>
<td>25</td>
<td>0.2389</td>
</tr>
<tr>
<td>7</td>
<td>30</td>
<td>0.3673</td>
</tr>
<tr>
<td>8</td>
<td>35</td>
<td>0.2864</td>
</tr>
<tr>
<td>9</td>
<td>40</td>
<td>0.3248</td>
</tr>
<tr>
<td>10</td>
<td>45</td>
<td>0.1987</td>
</tr>
<tr>
<td>11</td>
<td>50</td>
<td>0.1915</td>
</tr>
<tr>
<td>12</td>
<td>55</td>
<td>0.2685</td>
</tr>
<tr>
<td>13</td>
<td>60</td>
<td>0.3241</td>
</tr>
<tr>
<td>14</td>
<td>65</td>
<td>0.1995</td>
</tr>
<tr>
<td>15</td>
<td>70</td>
<td>0.1937</td>
</tr>
<tr>
<td>16</td>
<td>75</td>
<td>0.2643</td>
</tr>
<tr>
<td>17</td>
<td>80</td>
<td>0.1990</td>
</tr>
<tr>
<td>18</td>
<td>85</td>
<td>0.2467</td>
</tr>
<tr>
<td>19</td>
<td>90</td>
<td>0.3409</td>
</tr>
<tr>
<td>20</td>
<td>95</td>
<td>0.3212</td>
</tr>
<tr>
<td>21</td>
<td>100</td>
<td>0.2355</td>
</tr>
</tbody>
</table>

Table 6.2 Time taken to Download files in (KB) using AES encryption
Table 6.3 Time taken to upload files in (KB) using RSA encryption

<table>
<thead>
<tr>
<th>User Data</th>
<th>File Size(KB)</th>
<th>Time Required for Uploading /sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0.3239</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>0.4861</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>0.5181</td>
</tr>
<tr>
<td>4</td>
<td>15</td>
<td>0.5491</td>
</tr>
<tr>
<td>5</td>
<td>20</td>
<td>0.5754</td>
</tr>
<tr>
<td>6</td>
<td>25</td>
<td>0.5814</td>
</tr>
<tr>
<td>7</td>
<td>30</td>
<td>0.5899</td>
</tr>
<tr>
<td>8</td>
<td>35</td>
<td>0.5940</td>
</tr>
<tr>
<td>9</td>
<td>40</td>
<td>0.6011</td>
</tr>
<tr>
<td>10</td>
<td>45</td>
<td>0.6098</td>
</tr>
<tr>
<td>11</td>
<td>50</td>
<td>0.6173</td>
</tr>
<tr>
<td>12</td>
<td>55</td>
<td>0.6259</td>
</tr>
<tr>
<td>13</td>
<td>60</td>
<td>0.6316</td>
</tr>
<tr>
<td>14</td>
<td>65</td>
<td>0.6422</td>
</tr>
<tr>
<td>15</td>
<td>70</td>
<td>0.6684</td>
</tr>
<tr>
<td>16</td>
<td>75</td>
<td>0.6841</td>
</tr>
<tr>
<td>17</td>
<td>80</td>
<td>0.6904</td>
</tr>
<tr>
<td>18</td>
<td>85</td>
<td>0.7029</td>
</tr>
<tr>
<td>19</td>
<td>90</td>
<td>0.7336</td>
</tr>
<tr>
<td>20</td>
<td>95</td>
<td>0.7758</td>
</tr>
<tr>
<td>21</td>
<td>100</td>
<td>0.8050</td>
</tr>
<tr>
<td>User Data</td>
<td>File Size(KB)</td>
<td>Time Required for Downloading /sec</td>
</tr>
<tr>
<td>-----------</td>
<td>---------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RSA</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0.1838</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>0.6040</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>1.0555</td>
</tr>
<tr>
<td>4</td>
<td>15</td>
<td>1.6727</td>
</tr>
<tr>
<td>5</td>
<td>20</td>
<td>2.0014</td>
</tr>
<tr>
<td>6</td>
<td>25</td>
<td>2.4284</td>
</tr>
<tr>
<td>7</td>
<td>30</td>
<td>2.9826</td>
</tr>
<tr>
<td>8</td>
<td>35</td>
<td>3.4127</td>
</tr>
<tr>
<td>9</td>
<td>40</td>
<td>3.9675</td>
</tr>
<tr>
<td>10</td>
<td>45</td>
<td>4.3222</td>
</tr>
<tr>
<td>11</td>
<td>50</td>
<td>4.9210</td>
</tr>
<tr>
<td>12</td>
<td>55</td>
<td>5.4399</td>
</tr>
<tr>
<td>13</td>
<td>60</td>
<td>5.9589</td>
</tr>
<tr>
<td>14</td>
<td>65</td>
<td>6.3603</td>
</tr>
<tr>
<td>15</td>
<td>70</td>
<td>6.7618</td>
</tr>
<tr>
<td>16</td>
<td>75</td>
<td>7.2111</td>
</tr>
<tr>
<td>17</td>
<td>80</td>
<td>7.6889</td>
</tr>
<tr>
<td>18</td>
<td>85</td>
<td>8.0094</td>
</tr>
<tr>
<td>19</td>
<td>90</td>
<td>8.3098</td>
</tr>
<tr>
<td>20</td>
<td>95</td>
<td>8.5725</td>
</tr>
<tr>
<td>21</td>
<td>100</td>
<td>8.7650</td>
</tr>
</tbody>
</table>

Table 6.4 Time taken to Download files in (KB) using RSA encryption
5.1 Graphical representation

It can be seen from the graph Figure 16 that when data size is increased the graph steadily raises which shows the time taken gets more but when it reached a peak of 0.7 seconds there is a gradual fall which shows once the communication between the servers is steady the graph fell gradually and it went up and down widely. This graph shows that when the data file is uploaded it maintains a constant time ranging from (0.5 to 0.7) throughout the process.

Similarly from the graph Figure 17 shows that decrypting the text and downloading it from the Dropbox shows that AES maintains a constant time with less fluctuation.
It can be seen from the RSA graph Figure 18 that the line raised steadily when the file size increases. This shows that when the file size increases the time taken to upload data in dropbox will also increases gradually.
Figure 18

RSA UPLOAD

Time Required for Uploading /sec

FILE SIZE(KB)

0 20 40 60 80 100

RSA DOWNLOAD

Time Required for Downloading /sec

FILE SIZE(KB)

0 20 40 60 80 100

0 1 2 3 4 5 6 7 8 9
RSA vs AES Uploading:

In the above graph the red curve shows AES encryption and the blue curve shows the RSA encryption. As per figure 20 the red curve is observed to be constant for all file sizes whereas the blue curve increases as the file size increases.

Uploading:

In the above graph the red curve shows AES encryption and the blue curve shows the RSA encryption. As per figure 20 the red curve is observed to be constant for all file sizes whereas the blue curve increases as the file size increases.
RSA vs AES Downloading

Like Uploading, downloading reading from figure 21 also shows that AES encryption values are constant when compared with the RSA encryption results.

From the above comparison we can determine that the AES has better results in the perspective of time taken to upload and download when compared to the RSA as the values are stable enough to perform better in the real time.
Conclusion

Answer to the research problem

The above mentioned method eventually secures the data which is in transit by encrypting the file before uploading it in Dropbox. This gives freedom to the dropbox consumers to share any personal data in dropbox as the data in rest is secured by dropbox itself. This double encryption method of securing data benefits the customer to use dropbox and it also increases the usage of dropbox from the business perspective.

AES and RSA has been the best encryption methods for cloud storage services from different survey made and from the experimentation result it is clear that AES performs fast and better in real time.

This paper describes the various aspects of the cloud storage services and its security issues. Through this paper we proposed a new application which gives additional security to the existing security method that prevails in the dropbox. This also enables to share the data securely to any no: of users using the private key generated.

Growing need of the security has always been the top priority for any users and the method suggested above will give an extended security to transfer the text file which is encrypted using two different encryptions.

As per the above surveys when any document is encrypted before storing the information will eventually protect the overall data without being theft, when it is even in transit or when it is stored in storage services.

Future work

In future sure there will be an increase in the usage of cloud storage services especially Dropbox. When more users use to transfer data through these services, then more security is needed. We can add attachment like adding pdf and other .doc files in the above stated application in future. With this we can share any documents securely to any users. The encryption method used in this will give confidence in the users to use external application to secure their information. Through this method no: of users using dropbox will also increases and this will eventually increases the Dropbox businesses.

The encrypted way of sending and receiving the data is considered to be one of the safest methods, in addition to that concept tokenization is also an emerging technique used in cloud.

We can also use this application to share other files such as word and jpeg by using POI apache package. This is a open source API available in the market where we can export the content to the word file. By this we can also secure the PDF file in future.
References

[1]. “cloud computing learning”, Panagiotics Kalagnacos, Panagiotis Karamples, Department of Information Technology, Hellenic American University, Manchester, N.H. - U.S.A.


[4]. Shui Yu, Senior Member, IEEE, Yonghong Tian, Senior Member, IEEE;” Can We Beat DDoS Attacks in Clouds?.,


[7]. Shahed Latif; Subra Kumaraswamy; “Cloud Security and Privacy”, Published by oReilly Media, Inc., 2009

[8]. Rania Ele Sawy Abd EL Rahim, Mohamed; “Cloud computing security”, ISSR, Cairo 2012.

[9]. Dawn Song, Elaine Shi, and Ian Fischer; “Cloud Data Protection for the Masses” University of California, Berkeley Umesh Shankar, Google


[11]. “Two Layered Protection for Sensitive Data in Cloud”; Kamlesh Kumar Hingwe Dept. of Computer Science and Engineering National Institute of Technology Tiruchirappalli, India


[13]. Dave Shackleford “Cloud Tokenization: Why it might replace cloud encryption”, voodo Security


[16] Dawn Song, Elaine Shi, and Ian Fischer “Cloud data protection for the masses”; University of California, Berkeley; Umesh Shankar, Google

[17] Feng Zhao, Chao Li, Chun Feng Liu; “A cloud computing security solution based on fully homomorphic encryption”; STATE GRID ELECTRIC POWER RESEARCH INSTITUTE Ages Wealth World, China.


[21] https://www.dropbox.com/developers


[23] Giuseppe Ateniese, Randal Burns, Reza Curtmola, Reza Curtmola Provable Data Possession at Untrusted Stores;

[24] Mehul A.; Shah Ram Swaminathan; Mary Baker “Privacy-Preserving Audit and Extraction of Digital Contents”; HP Labs, @hp.com


47


APPENDIX

Admin Creation
The admin creation is same as other sign up options in websites. Here we can register with our personal details and with the dropbox connection information.

/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package orutaprivacy;

import javax.swing.JOptionPane;

/**<*
 * @author Rajesh
 */
public class admin extends javax.swing.JFrame {

/**<*
 * Creates new form admin
 */
public admin() {
    initComponents();
}

/**<*
 * This method is called from within the constructor to initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is always
 * regenerated by the Form Editor.
 */
@ SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN: initComponents
private void initComponents() {
    JLabel1 = new javax.swing.JLabel();
    JLabel2 = new javax.swing.JLabel();
    uname_txt = new javax.swing.JTextField();
    JLabel3 = new javax.swing.JLabel();
    pass_txt = new javax.swing.JPasswordField();
    JLabel5 = new javax.swing.JLabel();
    JLabel6 = new javax.swing.JLabel();
    }
jLabel4 = new javax.swing.JLabel();

setDefaultCloseOpeneration(javax.swing.JFrame.EXIT_ON_CLOSE);
setResizable(false);
getContentPane().setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());

jLabel1.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N
jLabel1.setText("Username");
getContentPane().add(jLabel1, new
org.netbeans.lib.awtextra.AbsoluteConstraints(167, 106, 70, -1));

jLabel2.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N
jLabel2.setText("Password");
getContentPane().add(jLabel2, new
org.netbeans.lib.awtextra.AbsoluteConstraints(167, 161, 70, -1));

getContentPane().add(uname_txt, new
org.netbeans.lib.awtextra.AbsoluteConstraints(255, 103, 128, -1));

jLabel3.setFont(new java.awt.Font("Tahoma", 0, 16)); // NOI18N
jLabel3.setForeground(new java.awt.Color(102, 255, 255));
jLabel3.setText("Sign in to Continue Public Auditing");
getContentPane().add(jLabel3, new
org.netbeans.lib.awtextra.AbsoluteConstraints(96, 26, -1, -1));

getContentPane().add(pass_txt, new
org.netbeans.lib.awtextra.AbsoluteConstraints(255, 158, 128, -1));

jLabel5.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/inter/adminlogini.png"))); // NOI18N
jLabel5.addMouseListener(new java.awt.event.MouseAdapter() { public void mouseClicked(java.awt.event.MouseEvent evt) { 
jLabel5MouseClicked(evt);
}});
getContentPane().add(jLabel5, new
org.netbeans.lib.awtextra.AbsoluteConstraints(300, 230, -1, 20));

jLabel6.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/inter/resetimg123.png"))); // NOI18N
jLabel6.addMouseListener(new java.awt.event.MouseAdapter() { public void mouseClicked(java.awt.event.MouseEvent evt) { 
jLabel6MouseClicked(evt);
}});
getContentPane().add(jLabel6, new
org.netbeans.lib.awtextra.AbsoluteConstraints(210, 230, -1, 20));
jLabel4.setIcon(new javax.swing.ImageIcon(getClass().getResource("/inter/uploaimg.png"))); // NOI18N
getContentPane().add(jLabel4, new org.netbeans.lib.awtextra.AbsoluteConstraints(1, 0, 390, 310));

pack();
}// </editor-fold>//GEN-END:initComponents

private void jLabel5MouseClicked(java.awt.event.MouseEvent evt) {//GEN-FIRST:event_jLabel5MouseClicked
String user=uname_txt.getText();
String pass=pass_txt.getText();
if(user.isEmpty())
{
    return;
}
else
{
    if(user.equals("admin")&&pass.equals("123"))
    {
        new ThirdPartyOwner().setVisible(true);
        dispose();
    }
    else
    {
        JOptionPane.showMessageDialog(null,"Invalid Username or Password");
    }
    // TODO add your handling code here:
}//GEN-LAST:event_jLabel5MouseClicked

private void jLabel6MouseClicked(java.awt.event.MouseEvent evt) {//GEN-FIRST:event_jLabel6MouseClicked
uname_txt.setText(" ");
pass_txt.setText(" ");
// TODO add your handling code here:
}//GEN-LAST:event_jLabel6MouseClicked

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc="Look and feel setting code (optional) ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
     * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
     */
    try {

for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
    if ("Nimbus".equals(info.getName())) {
        javax.swing.UIManager.setLookAndFeel(info.getClassName());
        break;
    }
}
}
catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(admin.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
}
catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(admin.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
}
catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(admin.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
}
catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(admin.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
}
} //</editor-fold>

/* Create and display the form */
java.awt.EventQueue.invokeLater(new Runnable() {
    public void run() {
        new admin().setVisible(true);
    }
});

// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
private javax.swing.JLabel jLabel4;
private javax.swing.JLabel jLabel5;
private javax.swing.JLabel jLabel6;
private javax.swing.JPasswordField pass_txt;
private javax.swing.JTextField uname_txt;
// End of variables declaration//GEN-END:variables

AUTHORIZATION
package orutaprvacy;

import com.ecs.API.dropbox.Authorization;
import com.dropbox.core.DbxException;
import com.ecs.API.util.AppConfiguration;
import static com.ecs.API.util.AppConfiguration.CONF_APP_KEY;
import static com.ecs.API.util.AppConfiguration.CONF_APP_SECRET;
import static com.ecs.API.util.AppConfiguration.CONF_APP_AUTH;
import java.awt.Color;
import java.io.File;
import java.io.IOException;
import java.util.HashMap;
import java.util.Map;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;

public class Authorizer extends javax.swing.JFrame {

    private Authorization authorize;
    private AppConfiguration appConfig;

    /**
     * Creates new form Authorizer
     */
    public Authorizer() {
        initComponents();
        appConfig = new AppConfiguration();
        if(!appConfig.loadConfiguration()){
            appConfig.createConf();
        }
    }

    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    @Override
    public void initComponents() {
        // TODO: custom initialize components
    }

}
// <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
private void initComponents() {

    jPanel1 = new javax.swing.JPanel();
    jLabel1 = new javax.swing.JLabel();
    jPanel2 = new javax.swing.JPanel();
    jLabel2 = new javax.swing.JLabel();
    jLabel3 = new javax.swing.JLabel();
    txtAppKey = new javax.swing.JTextField();
    txtAppSecret = new javax.swing.JTextField();
    btnURLGen = new javax.swing.JButton();
    jPanel4 = new javax.swing.JPanel();
    txtAuthCode = new javax.swing.JTextField();
    btnValidate = new javax.swing.JButton();
    jScrollPane1 = new javax.swing.JScrollPane();
    txtarURL = new javax.swing.JTextArea();
    jLabel8 = new javax.swing.JLabel();
    btnSaveAuth = new javax.swing.JButton();
    lblStatus = new javax.swing.JLabel();

    setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
    setTitle("Dropbox Authorization");
    setAlwaysOnTop(true);
    setResizable(false);

    jPanel1.setBorder(javax.swing.BorderFactory.createEtchedBorder());

    jLabel1.setFont(new java.awt.Font("Ubuntu", 1, 24)); // NOI18N
    jLabel1.setText("DropBox Authorization");

    jPanel2.setBorder(javax.swing.BorderFactory.createTitledBorder(null,
"Application Info", javax.swing.border.TitledBorder.LEFT,
javax.swing.border.TitledBorder.TOP));

    jLabel2.setFont(new java.awt.Font("Ubuntu", 0, 16)); // NOI18N
    jLabel2.setText("App Key");
    jLabel3.setFont(new java.awt.Font("Ubuntu", 0, 16)); // NOI18N
    jLabel3.setText("App Secret");

    btnURLGen.setText("Authorize URL");
    btnURLGen.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnURLGenActionPerformed(evt);

            }
    });

    jPanel4.setBorder(javax.swing.BorderFactory.createTitledBorder(null,
"Application Info", javax.swing.border.TitledBorder.LEFT,
javax.swing.border.TitledBorder.TOP));

    txtAuthCode.setEditable(false); // NOI18N
    txtarURL.setEditable(true); // NOI18N

    jScrollPane1.setViewportView(txtarURL); // NOI18N

    btnSaveAuth.setText("Save Authorization");

    setVisible(true); // NOI18N

    setCursor(new java.awt.Cursor(java.awt.Cursor.DEFAULT_CURSOR)); // NOI18N
    setFont(new java.awt.Font("Ubuntu", 0, 12)); // NOI18N
    setPreferredSize(new java.awt.Dimension(480, 360)); // NOI18N

    pack(); // NOI18N
//GEN-END:initComponents
}
javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);
jPanel2.setLayout(jPanel2Layout);
jPanel2Layout.setHorizontalGroup(
    jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jPanel2Layout.createSequentialGroup()
        .addGap(150, 150, 150)
        .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE,
                javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE,
                javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
        .addGap(29, 29, 29)
        .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
            .addComponent(txtAppSecret, javax.swing.GroupLayout.DEFAULT_SIZE, 201, Short.MAX_VALUE)
            .addComponent(txtAppKey))
    .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
    .addGroup(jPanel2Layout.createSequentialGroup()
        .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        .addComponent(btnURLGen)
    .addContainerGap())
    jPanel2Layout.setVerticalGroup(
        jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel2Layout.createSequentialGroup()
            .addGap(23, 23, 23)
            .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE,
                    javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(txtAppKey))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT_SIZE,
                javax.swing.GroupLayout.PREFERRED_SIZE)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT_SIZE,
                javax.swing.GroupLayout.PREFERRED_SIZE)
            .addComponent(btnURLGen)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT_SIZE,
                javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGap(23, 23, 23))
        .addComponent(txtAppSecret, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
            javax.swing.GroupLayout.PREFERRED_SIZE)
    .addContainerGap()});

jPanel2Layout.setHorizontalGroup(
    jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jPanel2Layout.createSequentialGroup()
        .addGap(23, 23, 23)
        .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
            .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE,
                javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addComponent(txtAppKey))
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT_SIZE,
            javax.swing.GroupLayout.PREFERRED_SIZE)
        .addComponent(btnURLGen)
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT_SIZE,
            javax.swing.GroupLayout.PREFERRED_SIZE)
        .addGap(23, 23, 23))
    .addGroup(jPanel2Layout.createSequentialGroup()
        .addGap(23, 23, 23)
        .addComponent(txtAppSecret, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
            javax.swing.GroupLayout.PREFERRED_SIZE)
    .addContainerGap()});

jPanel2Layout.setHorizontalGroup(
    jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jPanel2Layout.createSequentialGroup()
        .addGap(23, 23, 23)
        .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
            .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE,
                javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addComponent(txtAppKey))
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT_SIZE,
            javax.swing.GroupLayout.PREFERRED_SIZE)
        .addComponent(btnURLGen)
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT_SIZE,
            javax.swing.GroupLayout.PREFERRED_SIZE)
        .addGap(23, 23, 23))
    .addGroup(jPanel2Layout.createSequentialGroup()
        .addGap(23, 23, 23)
        .addComponent(txtAppSecret, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
            javax.swing.GroupLayout.PREFERRED_SIZE)
    .addContainerGap()});
javax.swing.GroupLayout.PREFERRED_SIZE, 30,
javax.swing.GroupLayout.PREFERRED_SIZE))
.addGap(18, 18, 18)
BASELINE)
    .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
.addGap(18, 18, 18)
.addComponent(txtAppSecret,
javax.swing.GroupLayout.PREFERRED_SIZE, 30,
javax.swing.GroupLayout.PREFERRED_SIZE))
BASELINE)
    .addComponent(inValiderrorCode, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE));
    jPanel4.setBorder(javax.swing.BorderFactory.createTitledBorder(null, "URL
Authorization", javax.swing.border.TitledBorder.LEFT,
javax.swing.border.TitledBorder.TOP));
btnValidate.setText("Validate Code");
btnValidate.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnValidateActionPerformed(evt);
    }
});
txtarURL.setColumns(20);
txtarURL.setRows(5);
jScrollPane1.setViewportView(txtarURL);

jLabel8.setText("Authorization Code");
btnSaveAuth.setText("Save Auth");
btnSaveAuth.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnSaveAuthActionPerformed(evt);
    }
});
javax.swing.GroupLayout jPanel4Layout = new
javax.swing.GroupLayout(jPanel4Layout = new
javax.swing.GroupLayout(jPanel4Layout);
jPanel4.setLayout(jPanel4Layout);
jPanel4Layout.setHorizontalGroup(
jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jPanel4Layout.createSequentialGroup()
    .addGap(18, 18, 18)
    .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
    .addGap(18, 18, 18)
    .addComponent(txtAppSecret,
javax.swing.GroupLayout.PREFERRED_SIZE, 30,
javax.swing.GroupLayout.PREFERRED_SIZE)
    .addGap(18, 18, 18)
    .addComponent(btnURLGen)
    .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE));
    jPanel4.setBorder(javax.swing.BorderFactory.createTitledBorder(null, "URL
Authorization", javax.swing.border.TitledBorder.LEFT,
javax.swing.border.TitledBorder.TOP));
btnValidate.setText("Validate Code");
btnValidate.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnValidateActionPerformed(evt);
    }
});
txtarURL.setColumns(20);
txtarURL.setRows(5);
jScrollPane1.setViewportView(txtarURL);

jLabel8.setText("Authorization Code");
btnSaveAuth.setText("Save Auth");
btnSaveAuth.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnSaveAuthActionPerformed(evt);
    }
});
javax.swing.GroupLayout jPanel4Layout = new
javax.swing.GroupLayout(jPanel4Layout = new
javax.swing.GroupLayout(jPanel4Layout);
jPanel4.setLayout(jPanel4Layout);
jPanel4Layout.setHorizontalGroup(
jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jPanel4Layout.createSequentialGroup()
    .addGap(18, 18, 18)
    .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)}
    .addGap(18, 18, 18)
    .addComponent(txtAppSecret,
javax.swing.GroupLayout.PREFERRED_SIZE, 30,
javax.swing.GroupLayout.PREFERRED_SIZE)
    .addGap(18, 18, 18)
    .addComponent(btnURLGen)
    .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE));
    jPanel4.setBorder(javax.swing.BorderFactory.createTitledBorder(null, "URL
Authorization", javax.swing.border.TitledBorder.LEFT,
javax.swing.border.TitledBorder.TOP));
btnValidate.setText("Validate Code");
btnValidate.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnValidateActionPerformed(evt);
    }
});
txtarURL.setColumns(20);
txtarURL.setRows(5);
jScrollPane1.setViewportView(txtarURL);

jLabel8.setText("Authorization Code");
btnSaveAuth.setText("Save Auth");
btnSaveAuth.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnSaveAuthActionPerformed(evt);
    }
});
javax.swing.GroupLayout jPanel4Layout = new
javax.swing.GroupLayout(jPanel4Layout = new
javax.swing.GroupLayout(jPanel4Layout);
jPanel4.setLayout(jPanel4Layout);
jPanel4Layout.setHorizontalGroup(
jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jPanel4Layout.createSequentialGroup()
    .addGap(18, 18, 18)
    .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE));
    jPanel4.setBorder(javax.swing.BorderFactory.createTitledBorder(null, "URL
Authorization", javax.swing.border.TitledBorder.LEFT,
javax.swing.border.TitledBorder.TOP));
btnValidate.setText("Validate Code");
btnValidate.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnValidateActionPerformed(evt);
    }
});
txtarURL.setColumns(20);
txtarURL.setRows(5);
jScrollPane1.setViewportView(txtarURL);

jLabel8.setText("Authorization Code");
btnSaveAuth.setText("Save Auth");
btnSaveAuth.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnSaveAuthActionPerformed(evt);
    }
});
javax.swing.GroupLayout jPanel4Layout = new
javax.swing.GroupLayout(jPanel4Layout = new
javax.swing.GroupLayout(jPanel4Layout);
jPanel4.setLayout(jPanel4Layout);
jPanel4Layout.setHorizontalGroup(
jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jPanel4Layout.createSequentialGroup()
    .addGap(18, 18, 18)
    .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE));
    jPanel4.setBorder(javax.swing.BorderFactory.createTitledBorder(null, "URL
Authorization", javax.swing.border.TitledBorder.LEFT,
javax.swing.border.TitledBorder.TOP));
btnValidate.setText("Validate Code");
btnValidate.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnValidateActionPerformed(evt);
    }
});
txtarURL.setColumns(20);
txtarURL.setRows(5);
jScrollPane1.setViewportView(txtarURL);

jLabel8.setText("Authorization Code");
btnSaveAuth.setText("Save Auth");
btnSaveAuth.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnSaveAuthActionPerformed(evt);
    }
});
javax.swing.GroupLayout jPanel4Layout = new
javax.swing.GroupLayout(jPanel4Layout = new
javax.swing.GroupLayout(jPanel4Layout);
jPanel4.setLayout(jPanel4Layout);
jPanel4Layout.setHorizontalGroup(
jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jPanel4Layout.createSequentialGroup()
    .addGap(18, 18, 18)
    .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE));
    jPanel4.setBorder(javax.swing.BorderFactory.createTitledBorder(null, "URL
Authorization", javax.swing.border.TitledBorder.LEFT,
javax.swing.border.TitledBorder.TOP));
btnValidate.setText("Validate Code");
btnValidate.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnValidateActionPerformed(evt);
    }
});
txtarURL.setColumns(20);
txtarURL.setRows(5);
jScrollPane1.setViewportView(txtarURL);

jLabel8.setText("Authorization Code");
btnSaveAuth.setText("Save Auth");
btnSaveAuth.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnSaveAuthActionPerformed(evt);
    }
});
javax.swing.GroupLayout jPanel4Layout = new
javax.swing.GroupLayout(jPanel4Layout = new
javax.swing.GroupLayout(jPanel4Layout);
jPanel4.setLayout(jPanel4Layout);
jPanel4Layout.setHorizontalGroup(
jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jPanel4Layout.createSequentialGroup()
    .addGap(18, 18, 18)
    .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE));
    jPanel4.setBorder(javax.swing.BorderFactory.createTitledBorder(null, "URL
Authorization", javax.swing.border.TitledBorder.LEFT,
javax.swing.border.TitledBorder.TOP));
btnValidate.setText("Validate Code");
btnValidate.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnValidateActionPerformed(evt);
    }
});
txtarURL.setColumns(20);
txtarURL.setRows(5);
jScrollPane1.setViewportView(txtarURL);

jLabel8.setText("Authorization Code");
btnSaveAuth.setText("Save Auth");
btnSaveAuth.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnSaveAuthActionPerformed(evt);
    }
});
javax.swing.GroupLayout jPanel4Layout = new
javax.swing.GroupLayout(jPanel4Layout = new
javax.swing.GroupLayout(jPanel4Layout);
jPanel4.setLayout(jPanel4Layout);
jPanel4Layout.setHorizontalGroup(
jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jPanel4Layout.createSequentialGroup()
    .addGap(18, 18, 18)
    .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE));
    jPanel4.setBorder(javax.swing.BorderFactory.createTitledBorder(null, "URL
Authorization", javax.swing.border.TitledBorder.LEFT,
javax.swing.border.TitledBorder.TOP));
btnValidate.setText("Validate Code");
btnValidate.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnValidateActionPerformed(evt);
    }
});
txtarURL.setColumns(20);
txtarURL.setRows(5);
jScrollPane1.setViewportView(txtarURL);

jLabel8.setText("Authorization Code");
btnSaveAuth.setText("Save Auth");
btnSaveAuth.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnSaveAuthActionPerformed(evt);
    }
});
javax.swing.GroupLayout jPanel4Layout = new
javax.swing.GroupLayout(jPanel4Layout = new
javax.swing.GroupLayout(jPanel4Layout);
jPanel4.setLayout(jPanel4Layout);
jPanel4Layout.setHorizontalGroup(
jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jPanel4Layout.createSequentialGroup()
    .addGap(18, 18, 18)
    .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE));
    jPanel4.setLayout(jPanel4Layout);
jPanel4Layout.setHorizontaLGroup(
jPanel4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
jPanel1.setLayout(jPanel1Layout);
jPanel1Layout.setHorizontalGroup(
    jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jPanel1Layout.createSequentialGroup()
    .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addComponent(jLabel1)
    .addGroup(jPanel1Layout.createSequentialGroup()
    .addContainerGap()
    .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addComponent(lblStatus, javax.swing.GroupLayout.Alignment.TRAILING)
    .addComponent(jPanel4, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))))
    .addContainerGap())
    jPanel1Layout.setVerticalGroup(
    jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jPanel1Layout.createSequentialGroup()
    .addComponent(jLabel1)
    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
    .addComponent(jPanel4, javax.swing.GroupLayout.PREFERRED_SIZE)
    .addGap(0, 0, Short.MAX_VALUE))
    .addContainerGap())
);
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
        .addGroup(layout.createSequentialGroup()
            .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
                javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .addComponent(lblStatus, javax.swing.GroupLayout.DEFAULT_SIZE, 23,
                Short.MAX_VALUE)
            .addContainerGap());

javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addContainerGap()
            .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
                javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .addContainerGap())
    layout.setVerticalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
                    javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addContainerGap())
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
                    javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addContainerGap())
    );

pack();
} //</editor-fold>//GEN-END: initComponents

private void btnURLGenActionPerformed(java.awt.event.ActionEvent evt) {
    //GEN-FIRST:event_btnURLGenActionPerformed
    String appKey = txtAppKey.getText().trim();
    String appSecret = txtAppSecret.getText().trim();
    if (!appKey.isEmpty() && !appSecret.isEmpty()) {
        authorize = new Authorization(appKey, appSecret);
        // Storing App Values to Config
        Map<String, String> properties = new HashMap<>();
        properties.put(CONF_APP_KEY, appKey);
        properties.put(CONF_APP_SECRET, appSecret);
        appConfig.storeProperties(properties);

        txtarURL.setText(authorize.getAuthURL());
        lblStatus.setForeground(Color.blue);
        lblStatus.setText("Authorization URL generated Successfully!");
    }
} //GEN-LAST:event_btnURLGenActionPerformed
try {
    authorize.generateAuthCode();
} catch (IOException ex) {
    JOptionPane.showMessageDialog(rootPane, "Opening Browser and navigating URI failed... Please Validate URI Manually", "Failure", JOptionPane.INFORMATION_MESSAGE);
} else {
    JOptionPane.showMessageDialog(rootPane, "App Key and App Secret cannot be left Empty", "Enter Values", JOptionPane.ERROR_MESSAGE);
    lblStatus.setForeground(Color.RED);
    lblStatus.setText("App Key and App Secret cannot be left Empty!");
}

private void btnValidateActionPerformed(java.awt.event.ActionEvent evt) {
    String appAccessCode = txtAuthCode.getText().trim();
    if (!appAccessCode.isEmpty()) {
        try {
            authorize.authorize(appAccessCode);
            lblStatus.setForeground(Color.blue);
            lblStatus.setText("Access Code Validated Successfully!");
        } catch (DbxException ex) {
            ex.printStackTrace();
            JOptionPane.showMessageDialog(rootPane, ex.getMessage(), "Error in Validation", JOptionPane.ERROR_MESSAGE);
            lblStatus.setForeground(Color.RED);
            lblStatus.setText("Error in Validation!");
        }
    } else {
        JOptionPane.showMessageDialog(rootPane, "App Access Code cannot be left Empty", "Enter Access code", JOptionPane.ERROR_MESSAGE);
        lblStatus.setForeground(Color.RED);
        lblStatus.setText("App Access Code cannot be left Empty!");
    }
}

private void btnSaveAuthActionPerformed(java.awt.event.ActionEvent evt) {
    JFileChooser saveDialog = new JFileChooser();
    saveDialog.setDialogTitle("Specify a file to save");
    int value = saveDialog.showSaveDialog(rootPane);
    if (value == JFileChooser.APPROVE_OPTION) {
        File fileToSave = saveDialog.getSelectedFile();
        try {
            authorize.saveAuthInfo(fileToSave);
        }
    }
}
// Storing Auth File Name
Map<String, String> properties = new HashMap<>();
properties.put(CONF_APP_AUTH, fileToSave.getAbsolutePath());
appConfig.storeProperties(properties);

lblStatus.setForeground(Color.blue);
lblStatus.setText("App Access Token Saved Successfully!");
} catch (IOException ex) {
    JOptionPane.showMessageDialog(rootPane, "App Access Token cannot be
    Saved", "Error Saving Access Token", JOptionPane.ERROR_MESSAGE);
    lblStatus.setForeground(Color.RED);
lblStatus.setText("App Access Token cannot be Saved!");
}
} else if (value == JFileChooser.ERROR_OPTION) {
    JOptionPane.showMessageDialog(rootPane, "App Access Token cannot be
    Saved", "Error Saving Access Token", JOptionPane.ERROR_MESSAGE);
    lblStatus.setForeground(Color.RED);
lblStatus.setText("App Access Token cannot be Saved!");
} else {
    //JOptionPane.showMessageDialog(rootPane, "App Access Token Save
cancelled", "Error Saving Access Token", JOptionPane.ERROR_MESSAGE);
    lblStatus.setForeground(Color.RED);
lblStatus.setText("App Access Token Save cancelled!");
}
} //GEN-LAST:event_btnSaveAuthActionPerformed

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc="Look and feel setting code
    (optional) ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look
    and feel. *
    * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
    */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info :
            javax.swing.UIManager.getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
        
    }
    } catch (ClassNotFoundException ex) {

/* Create and display the form */
java.awt.EventQueue.invokeLater(new Runnable() {
    public void run() {
        new Authorizer().setVisible(true);
    }
});

// Variables declaration - do not modify
private javax.swing.JButton btnSaveAuth;
private javax.swing.JButton btnURLGen;
private javax.swing.JButton btnValidate;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
private javax.swing.JLabel jLabel8;
private javax.swing.JPanel jPanel1;
private javax.swing.JPanel jPanel2;
private javax.swing.JPanel jPanel4;
private javax.swing.JScrollPane jScrollPane1;
private javax.swing.JLabel lblStatus;
private javax.swing.JTextField txtAppKey;
private javax.swing.JTextField txtAppSecret;
private javax.swing.JTextField txtAuthCode;
private javax.swing.JTextArea txtarURL;

DATA OWNER LOGIN
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package orutaprivacy;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;

/**
 * @author Rajesh
 */
public class dataownerlogin extends javax.swing.JFrame {
    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {
        jLabel2 = new javax.swing.JLabel();
        jLabel1 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        jLabel5 = new javax.swing.JLabel();
        jLabel6 = new javax.swing.JLabel();
        jLabel7 = new javax.swing.JLabel();
        jLabel8 = new javax.swing.JLabel();
        jLabel9 = new javax.swing.JLabel();
        lbEmail = new javax.swing.JLabel();
        pno_txt = new javax.swing.JTextField();
        fname_txt = new javax.swing.JTextField();
        lname_txt = new javax.swing.JTextField();
        uname_txt = new javax.swing.JTextField();
    }
    // </editor-fold>
    private void initComponents() {
        jLabel2 = new javax.swing.JLabel();
        jLabel1 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        jLabel5 = new javax.swing.JLabel();
        jLabel6 = new javax.swing.JLabel();
        jLabel7 = new javax.swing.JLabel();
        jLabel8 = new javax.swing.JLabel();
        jLabel9 = new javax.swing.JLabel();
        lbEmail = new javax.swing.JLabel();
        pno_txt = new javax.swing.JTextField();
        fname_txt = new javax.swing.JTextField();
        lname_txt = new javax.swing.JTextField();
        uname_txt = new javax.swing.JTextField();
    }
pass_txt = new javax.swing.JTextField();
tfEmail = new javax.swing.JTextField();
bdate_txt = new javax.swing.JTextField();
gender_txt = new javax.swing.JTextField();
city_txt = new javax.swing.JTextField();
submitimg = new javax.swing.JLabel();
resetimg = new javax.swing.JLabel();
jLabel12 = new javax.swing.JLabel();
jLabel13 = new javax.swing.JLabel();
backgroundimg = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
setResizable(false);
getContentPane().setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());

jLabel2.setFont(new java.awt.Font("Times New Roman", 0, 23)); // NOI18N
jLabel2.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
jLabel2.setText("Create your Cloud Account");
getContentPane().add(jLabel2, new
org.netbeans.lib.awtextra.AbsoluteConstraints(0, 40, 680, -1));

jLabel1.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N
jLabel1.setText("Gender");
getContentPane().add(jLabel1, new
org.netbeans.lib.awtextra.AbsoluteConstraints(70, 330, 70, -1));

jLabel3.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N
jLabel3.setText("Firstname");
getContentPane().add(jLabel3, new
org.netbeans.lib.awtextra.AbsoluteConstraints(70, 130, 70, -1));

jLabel4.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N
jLabel4.setText("Lastname");
getContentPane().add(jLabel4, new
org.netbeans.lib.awtextra.AbsoluteConstraints(70, 170, 70, -1));

jLabel5.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N
jLabel5.setText("Username");
getContentPane().add(jLabel5, new
org.netbeans.lib.awtextra.AbsoluteConstraints(70, 210, 70, -1));

jLabel6.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N
jLabel6.setText("Password");
getContentPane().add(jLabel6, new
org.netbeans.lib.awtextra.AbsoluteConstraints(70, 250, 70, -1));

jLabel7.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N
jLabel7.setText("Birthdate");
getContentPane().add(jLabel7, new
org.netbeans.lib.awtextra.AbsoluteConstraints(70, 290, 70, -1));

jLabel8.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N
jLabel8.setText("Phoneno");
getContentPane().add(jLabel8, new
org.netbeans.lib.awtextra.AbsoluteConstraints(70, 410, 70, -1));

lbEmail.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N
lbEmail.setText("Email");
getContentPane().add(lbEmail, new
org.netbeans.lib.awtextra.AbsoluteConstraints(70, 450, 70, -1));

jLabel9.setFont(new java.awt.Font("Tahoma", 0, 14)); // NOI18N
jLabel9.setText("City");
getContentPane().add(jLabel9, new
org.netbeans.lib.awtextra.AbsoluteConstraints(70, 370, 70, -1));

pno_txt.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        pno_txtActionPerformed(evt);
    }
});
getContentPane().add(pno_txt, new
org.netbeans.lib.awtextra.AbsoluteConstraints(190, 400, 150, 30));
//
tfEmail.setText("email");
getContentPane().add(tfEmail, new
org.netbeans.lib.awtextra.AbsoluteConstraints(190, 440, 150, 30));

fname_txt.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        fname_txtActionPerformed(evt);
    }
});
getContentPane().add(fname_txt, new
org.netbeans.lib.awtextra.AbsoluteConstraints(190, 120, 150, 30));

lname_txt.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        lname_txtActionPerformed(evt);
    }
});
getContentPane().add(lname_txt, new
org.netbeans.lib.awtextra.AbsoluteConstraints(190, 160, 150, 30));

uname_txt.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        uname_txtActionPerformed(evt);
    }
});
getContentPane().add(uname_txt, new
org.netbeans.lib.awtextra.AbsoluteConstraints(190, 200, 150, 30));

pass_txt.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        pass_txtActionPerformed(evt);
    }
});
getContentPane().add(pass_txt, new
org.netbeans.lib.awtextra.AbsoluteConstraints(190, 240, 150, 30));
org.netbeans.lib awtextra.AbsoluteConstraints(190, 240, 150, 30));

        bdate_txt.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                bdate_txtActionPerformed(evt);
            }
        });
        getContentPane().add(bdate_txt, new
        org.netbeans.lib awtextra.AbsoluteConstraints(190, 280, 150, 30));

        gender_txt.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                gender_txtActionPerformed(evt);
            }
        });
        getContentPane().add(gender_txt, new
        org.netbeans.lib awtextra.AbsoluteConstraints(190, 320, 150, 30));

        city_txt.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                city_txtActionPerformed(evt);
            }
        });
        getContentPane().add(city_txt, new
        org.netbeans.lib awtextra.AbsoluteConstraints(190, 360, 150, 30));

        submitimg.setIcon(new javax.swing.ImageIcon(getClass().getResource("/inter/submitimag.png"))); //</NOI18N
        submitimg.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                submitimgMouseClicked(evt);
            }
        });
        getContentPane().add(submitimg, new
        org.netbeans.lib awtextra.AbsoluteConstraints(190, 400, 150, 30));

        resetimg.setIcon(new javax.swing.ImageIcon(getClass().getResource("/inter/resetimg123.png"))); </NOI18N
        resetimg.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                resetimgMouseClicked(evt);
            }
        });
        getContentPane().add(resetimg, new
        org.netbeans.lib awtextra.AbsoluteConstraints(190, 440, 150, 30));

        submitimg.setIcon(new javax.swing.ImageIcon(getClass().getResource("/inter/submitimag.png"))); </NOI18N
        submitimg.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                submitimgMouseClicked(evt);
            }
        });
        getContentPane().add(submitimg, new
        org.netbeans.lib awtextra.AbsoluteConstraints(190, 480, 150, 30));

        resetimg.setIcon(new javax.swing.ImageIcon(getClass().getResource("/inter/resetimg123.png"))); </NOI18N
        resetimg.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                resetimgMouseClicked(evt);
            }
        });
        getContentPane().add(resetimg, new
        org.netbeans.lib awtextra.AbsoluteConstraints(190, 520, 150, 30));

        submitimg.setIcon(new javax.swing.ImageIcon(getClass().getResource("/inter/submitimag.png"))); </NOI18N
        submitimg.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                submitimgMouseClicked(evt);
            }
        });
        getContentPane().add(submitimg, new
        org.netbeans.lib awtextra.AbsoluteConstraints(190, 560, 150, 30));

        resetimg.setIcon(new javax.swing.ImageIcon(getClass().getResource("/inter/resetimg123.png"))); </NOI18N
        resetimg.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                resetimgMouseClicked(evt);
            }
        });
        getContentPane().add(resetimg, new
        org.netbeans.lib awtextra.AbsoluteConstraints(190, 600, 150, 30));

        submitimg.setIcon(new javax.swing.ImageIcon(getClass().getResource("/inter/submitimag.png"))); </NOI18N
        submitimg.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                submitimgMouseClicked(evt);
            }
        });
        getContentPane().add(submitimg, new
        org.netbeans.lib awtextra.AbsoluteConstraints(190, 640, 150, 30));

        resetimg.setIcon(new javax.swing.ImageIcon(getClass().getResource("/inter/resetimg123.png"))); </NOI18N
        resetimg.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                resetimgMouseClicked(evt);
            }
        });
        getContentPane().add(resetimg, new
        org.netbeans.lib awtextra.AbsoluteConstraints(190, 680, 150, 30));

        submitimg.setIcon(new javax.swing.ImageIcon(getClass().getResource("/inter/submitimag.png"))); </NOI18N
        submitimg.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                submitimgMouseClicked(evt);
            }
        });
        getContentPane().add(submitimg, new
        org.netbeans.lib awtextra.AbsoluteConstraints(190, 720, 150, 30));

        resetimg.setIcon(new javax.swing.ImageIcon(getClass().getResource("/inter/resetimg123.png"))); </NOI18N
        resetimg.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                resetimgMouseClicked(evt);
            }
        });
        getContentPane().add(resetimg, new
        org.netbeans.lib awtextra.AbsoluteConstraints(190, 760, 150, 30));

        submitimg.setIcon(new javax.swing.ImageIcon(getClass().getResource("/inter/submitimag.png"))); </NOI18N
        submitimg.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                submitimgMouseClicked(evt);
            }
        });
        getContentPane().add(submitimg, new
        org.netbeans.lib awtextra.AbsoluteConstraints(190, 800, 150, 30));

        resetimg.setIcon(new javax.swing.ImageIcon(getClass().getResource("/inter/resetimg123.png"))); </NOI18N
        resetimg.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                resetimgMouseClicked(evt);
            }
        });
        getContentPane().add(resetimg, new
        org.netbeans.lib awtextra.AbsoluteConstraints(190, 840, 150, 30));

        submitimg.setIcon(new javax.swing.ImageIcon(getClass().getResource("/inter/submitimag.png"))); </NOI18N
        submitimg.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                submitimgMouseClicked(evt);
            }
        });
        getContentPane().add(submitimg, new
        org.netbeans.lib awtextra.AbsoluteConstraints(190, 880, 150, 30));

        resetimg.setIcon(new javax.swing.ImageIcon(getClass().getResource("/inter/resetimg123.png"))); </NOI18N
        resetimg.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                resetimgMouseClicked(evt);
            }
        });
        getContentPane().add(resetimg, new
        org.netbeans.lib awtextra.AbsoluteConstraints(190, 920, 150, 30));

        submitimg.setIcon(new javax.swing.ImageIcon(getClass().getResource("/inter/submitimag.png"))); </NOI18N
        submitimg.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                submitimgMouseClicked(evt);
            }
        });
        getContentPane().add(submitimg, new
        org.netbeans.lib awtextra.AbsoluteConstraints(190, 960, 150, 30));

        resetimg.setIcon(new javax.swing.ImageIcon(getClass().getResource("/inter/resetimg123.png"))); </NOI18N
        resetimg.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                resetimgMouseClicked(evt);
            }
        });
        getContentPane().add(resetimg, new
        org.netbeans.lib awtextra.AbsoluteConstraints(190, 1000, 150, 30));
jLabel12.setFont(new java.awt.Font("Times New Roman", 0, 18)); // NOI18N
jLabel12.setForeground(new java.awt.Color(102, 255, 255));
jLabel12.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
jLabel12.setText(ProjectConstants.TOPIC);
getContentPane().add(jLabel12, new
org.netbeans.lib.awtextra.AbsoluteConstraints(0, 0, 680, -1));

jLabel13.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/inter/backbtn.jpg"))); // NOI18N
jLabel13.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        jLabel13MouseClicked(evt);
    }
});
getContentPane().add(jLabel13, new
org.netbeans.lib.awtextras.AbsoluteConstraints(160, 490, 80, 30));

backgroundimg.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/image/99.jpg"))); // NOI18N
getContentPane().add(backgroundimg, new
org.netbeans.lib.awtextras.AbsoluteConstraints(0, 0, 680, 550));

pack();
} // </editor-fold>

private void submitimgMouseClicked(java.awt.event.MouseEvent evt) {
//new index().setVisible(true);
try
{
    Connection con=null;
    String url="jdbc:mysql://localhost:3306/";
    String driver="com.mysql.jdbc.Driver";
    String db="orutapreserving";
    String Firstname = fname_txt.getText();
    String Lastname = lname_txt.getText();
    String Username = uname_txt.getText();
    String Password = pass_txt.getText();
    String Birthdate = bdate_txt.getText();
    String Gender = gender_txt.getText();
    String City=city_txt.getText();
    String Phoneno = pno_txt.getText();
    String email = tfEmail.getText();
    Class.forName(driver);
    con=(Connection)DriverManager.getConnection(url+db, "root", "");

    String query = "INSERT INTO 
formreg(Firstname,Lastname,Username,Password,access_code,Birthdate,Gender,City

String query = "INSERT INTO 
formreg(Firstname,Lastname,Username,Password,access_code,Birthdate,Gender,City

,Phoneno, user_email) 
                  + " VALUES (?,?,?,?,?,?,?,?,?,?)";

                // create the mysql insert preparedstatement
                PreparedStatement preparedStmt = con.prepareStatement(query);
                preparedStmt.setString (1, Firstname);
                preparedStmt.setString (2, Lastname);
                preparedStmt.setString (3, Username);
                preparedStmt.setString(4, Password);
                preparedStmt.setInt    (5, 0);
                preparedStmt.setString (6, Birthdate);
                preparedStmt.setString (7, Gender);
                preparedStmt.setString (8, City);
                preparedStmt.setString(9, Phoneno);
                preparedStmt.setString(10, email);

                // execute the preparedstatement
                preparedStmt.execute();

                System.out.println(query);

                System.out.println("value inserted");
            }
        }
    }

    catch(Exception ex)
    {
        System.out.println(ex);
    }
    new index().setVisible(true);
    dispose();
    // TODO add your handling code here:
}

private void bdate_txtActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void resetimgMouseClicked(java.awt.event.MouseEvent evt) {
    fname_txt.setText(""");
    lname_txt.setText(""");
    uname_txt.setText(""");
    pass_txt.setText(""");
    bdate_txt.setText(""");
    gender_txt.setText(""");
    city_txt.setText(""");
    pno_txt.setText(""};
private void jLabel13MouseClicked(java.awt.event.MouseEvent evt) {
    new index().setVisible(true);
    dispose();
    // TODO add your handling code here:
}

private void fname_txtActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void gender_txtActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void pno_txtActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void city_txtActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void uname_txtActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc="Look and feel setting code (optional) ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look
     and feel.
     * For details see
     * http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
     */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    }
    catch (ClassNotFoundException | InstantiationException | IllegalAccessException | javax.swing.UnsupportedLookAndFeelException ex) {
        ex.printStackTrace();
    }
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    //</editor-fold>
    </
catch (ClassNotFoundException ex) {
    java.util.logging.Logger.getLogger(dataownerlogin.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
}

} catch (InstantiationException ex) {
    java.util.logging.Logger.getLogger(dataownerlogin.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
}

} catch (IllegalAccessException ex) {
    java.util.logging.Logger.getLogger(dataownerlogin.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
}

} catch (javax.swing.UnsupportedLookAndFeelException ex) {
    java.util.logging.Logger.getLogger(dataownerlogin.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
}

}  

//</editor-fold>

/* Create and display the form */
java.awt.EventQueue.invokeLater(new Runnable() {
    public void run() {
        new dataownerlogin().setVisible(true);
    }
});

}  

// Variables declaration - do not modify 
private javax.swing.JLabel backgroundimg;
private javax.swing.JLabel lbEmail;
private javax.swing.JTextField bdate_txt;
private javax.swing.JTextField city_txt;
private javax.swing.JTextField fname_txt;
private javax.swing.JTextField gender_txt;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel12;
private javax.swing.JLabel jLabel13;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
private javax.swing.JLabel jLabel4;
private javax.swing.JLabel jLabel5;
private javax.swing.JLabel jLabel6;
private javax.swing.JLabel jLabel7;
private javax.swing.JLabel jLabel8;
private javax.swing.JLabel jLabel9;
private javax.swing.JTextField lname_txt;
private javax.swing.JTextField pass_txt;
private javax.swing.JTextField pno_txt;
private javax.swing.JLabel resetimg;
private javax.swing.JLabel submitimg;
private javax.swing.JTextField uname_txt;
private javax.swing.JTextField tfEmail;

// End of variables declaration

Logic block
/**
 * @author rajesh
 */
public class LogicBlock {

/**
 * @author rajesh
 */
public LogicBlock() {

    System.out.println("HAI");

    int count = 63;
    int div = count / 5;
    int block = 1;
    int excess=count%5;

    List<String> b1 = new ArrayList<String>();
    List<String> b2 = new ArrayList<String>();
    List<String> b3 = new ArrayList<String>();
    List<String> b4 = new ArrayList<String>();
    List<String> b5 = new ArrayList<String>();

    for (int i = 0; i < count; i++) {
        for (int j = 0; j < div; j++) {
            switch (block) {
                case 1:
                    b1.add(String.valueOf(j));
                    b1.add(String.valueOf(j));
            }
        }
    }
}
break;
case 2:
    b2.add(String.valueOf(j));
break;
case 3:
    b3.add(String.valueOf(j));
break;
case 4:
    b4.add(String.valueOf(j));
break;
case 5:
    b5.add(String.valueOf(j));
    if(j==(div-1)) {
        System.out.println(count);
        for(int k=1; k<=excess;k++) {
            b5.add(String.valueOf(b5.size()+1));
        }
    }
break;
}

block++;

System.out.println(b1);
System.out.println(b2);
System.out.println(b3);
System.out.println(b4);
System.out.println(b5);

/**
 * @param args
 */
public static void main(String[] args) {
    LogicBlock s = new LogicBlock();
HOMOMORPHIC
package orutaprivacy;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.security.NoSuchAlgorithmException;
import java.util.Base64;
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.NoSuchPaddingException;
import javax.crypto.SecretKey;

public class HomomorphicEncryptionDecryptionAES {
    public static Cipher cipher;
    public static final String PUBLIC_KEY_FILE = "C:/encryptedSend/secret.key";
    private static SecretKey secretKey;

    public HomomorphicEncryptionDecryptionAES() {
        try {
            if (!areKeysPresent()) {
                // Method generates a pair of keys using the RSA algorithm and
                // stores it
                // in their respective files
                generateKeyFile();
            }
            cipher = Cipher.getInstance("AES");
            ObjectInputStream inputStream = null;
            FileInputStream fin = new FileInputStream(PUBLIC_KEY_FILE);
            // Encrypt the string using the public key
            inputStream = new ObjectInputStream(fin);
            secretKey = (SecretKey) inputStream.readObject();
            fin.close();
            inputStream.close();
        } catch (ClassNotFoundException | IOException | NoSuchAlgorithmException | NoSuchPaddingException e) {
            e.printStackTrace();
        }
    }

    // Encrypt the string using the public key
    inputFileStream = new FileInputStream(PUBLIC_KEY_FILE);

    try {
        // Method generates a pair of keys using the RSA algorithm and
        // stores it
        // in their respective files
        generateKeyFile();
    } catch (ClassNotFoundException | IOException | NoSuchAlgorithmException | NoSuchPaddingException e) {
        e.printStackTrace();
    }
}
public static boolean areKeysPresent() {
    File publicKey = new File(PUBLIC_KEY_FILE);
    if (publicKey.exists()) {
        return true;
    }
    return false;
}

private void generateKeyFile() {
    try {
        KeyGenerator keyGenerator = KeyGenerator.getInstance("AES");
        keyGenerator.init(128);
        SecretKey secretKey = keyGenerator.generateKey();
        File publicKeyFile = new File(PUBLIC_KEY_FILE);
        if (publicKeyFile.getParentFile() != null) {
            publicKeyFile.getParentFile().mkdirs();
        }
        publicKeyFile.createNewFile();
        FileOutputStream fout = new FileOutputStream(publicKeyFile);
        // Saving the Public key in a file
        ObjectOutputStream publicKeyOS = new ObjectOutputStream(fout);
        publicKeyOS.writeObject(secretKey);
        publicKeyOS.close();
        fout.close();
    } catch (NoSuchAlgorithmException | IOException e) {
        e.printStackTrace();
    }
}

public static void main(String[] args) throws Exception {
    HomomorphicEncryptionDecryptionAES start = new HomomorphicEncryptionDecryptionAES();
    HomomorphicEncryptionDecryptionAES end = new HomomorphicEncryptionDecryptionAES();
    String plainText = "This is for Rajesh 
for Research Article 
ORATA";
    System.out.println("Plain Text Before Encryption: " + plainText);
String encryptedText = start.encrypt(plainText);
System.out.println("Encrypted Text After Encryption: " + encryptedText);

String decryptedText = end.decrypt(encryptedText);
System.out.println("Decrypted Text After Decryption: " + decryptedText);
}

void temp() {
    try {
        KeyGenerator keyGenerator = KeyGenerator.getInstance("AES");
        keyGenerator.init(128);

        SecretKey secretKey = keyGenerator.generateKey();
cipher = Cipher.getInstance("AES");

        String plainText = "This is for Rajesh 
        for Research Article 
        ORATA";
System.out.println("Plain Text Before Encryption: " + plainText);

        String encryptedText = encrypt(plainText);
System.out.println("Encrypted Text After Encryption: "
 + encryptedText);

        String decryptedText = decrypt(encryptedText);
System.out.println("Decrypted Text After Decryption: "
 + decryptedText);
    } catch (Exception e) {
        e.printStackTrace();
    }
}

public String encrypt(String plainText) throws Exception {
    byte[] plainTextByte = plainText.getBytes();
cipher.init(Cipher.ENCRYPT_MODE, secretKey);
byte[] encryptedByte = cipher.doFinal(plainTextByte);
Base64.Encoder encoder = Base64.getEncoder();
String encryptedText = encoder.encodeToString(encryptedByte);
return encryptedText;
}

public String decrypt(String encryptedText) throws Exception {
    Base64.Decoder decoder = Base64.getDecoder();
byte[] encryptedTextByte = decoder.decode(encryptedText);
cipher.init(Cipher.DECRYPT_MODE, secretKey);
byte[] decryptedByte = cipher.doFinal(encryptedTextByte);
String decryptedText = new String(decryptedByte);
return decryptedText;
public class HomomorphicRSASecurity {
    private BigInteger n, d, e;
    private int bitlen = 1024;

    /** Create an instance that can encrypt using someone else's public key. */
    public HomomorphicRSASecurity(BigInteger newn, BigInteger newe) {
        n = newn;
        e = newe;
    }

    /** Create an instance that can both encrypt and decrypt. */
    public HomomorphicRSASecurity(int bits) {
        bitlen = bits;
        Random rs = new Random(1024);
        BigInteger p = new BigInteger(bitlen / 2, 100, rs);
        BigInteger q = new BigInteger(bitlen / 2, 100, rs);
        n = p.multiply(q);
        BigInteger m = (p.subtract(BigInteger.ONE)).multiply(q
                    .subtract(BigInteger.ONE));
        e = new BigInteger("3");
        while (m.gcd(e).intValue() > 1) {
            e = e.add(new BigInteger("2"));
        }
        d = e.modInverse(m);
    }

    /** Encrypt the given plaintext message. */
    public synchronized String encrypt(String message) {
        return (new BigInteger(message.getBytes())).modPow(e, n).toString();
    }

    public synchronized BigInteger encrypt(BigInteger message) {
        return message.modPow(e, n);
    }
}
return message.modPow(e, n);
}

/** Decrypt the given ciphertext message. */
public synchronized String decrypt(String message) {
    return new String((new BigInteger(message)).modPow(d, n).toByteArray());
}

/** Decrypt the given ciphertext message. */
public synchronized BigInteger decrypt(BigInteger message) {
    return message.modPow(d, n);
}

/** Generate a new public and private key set. */
public synchronized void generateKeys() {
    SecureRandom r = new SecureRandom();
    BigInteger p = new BigInteger(bitlen / 2, 100, r);
    BigInteger q = new BigInteger(bitlen / 2, 100, r);
    n = p.multiply(q);
    BigInteger m = (p.subtract(BigInteger.ONE)).multiply(q.subtract(BigInteger.ONE));
    e = new BigInteger("3");
    while (m.gcd(e).intValue() > 1) {
        e = e.add(new BigInteger("2"));
    }
    d = e.modInverse(m);
}

/** Return the modulus. */
public synchronized BigInteger getN() {
    return n;
}

/** Return the public key. */
public synchronized BigInteger getE() {
    return e;
}

/** Trivial test program. */
public static void main(String[] args) {
    HomomorphicRSASecurity rsaEncrypt = new HomomorphicRSASecurity(1024);
    HomomorphicRSASecurity rsaDecrypt = new HomomorphicRSASecurity(1024);

    String inputText = ProjectConstants.SAMPLE;
    System.out.println("Plaintext: " + inputText);
    BigInteger plaintext = new BigInteger(inputText.getBytes());
    BigInteger ciphertext = rsaEncrypt.encrypt(plaintext);
    System.out.println("Ciphertext: " + ciphertext);
plaintext = rsaDecrypt.decrypt(ciphertext);

String text2 = new String(plaintext.toByteArray());
System.out.println("Plaintext: " + text2);
}
Rajesh Vasudevan