

1. Introduction

With the considerable advancements in cloud computing, users and organizations are finding it increasingly appealing to store and share data through cloud services. Cloud service providers (such as Amazon, Microsoft, Apple, etc.) provide abundant cloud based services, ranging from small-scale personal services to large-scale industrial services. However, recent data breaches, such as releases of private photos, have raised concerns regarding the privacy of cloud-managed data. Actually, a cloud service provider is usually not secure due to design drawbacks of software and system vulnerability. As such, a critical issue is how to enforce data access control on the potentially untrusted cloud. In response to these security issues, numerous works have been proposed to support access control on untrusted cloud services by leveraging cryptographic primitives. Advanced cryptographic primitives are applied for enforcing many access control paradigms. For example, attribute-based encryption (ABE) is a cryptographic counterpart of attribute-based access control (ABAC) model. However, previous works mainly consider static scenarios in which access control policies rarely change. The previous works incur high overhead when access control policies need to be changed in practice. At a first glance, the revocation of a user's permission can be done by revoking his access to the keys with which the files are encrypted. This solution, however, is not secure as the user can keep a local copy of the keys before the revocation.

2. SYSTEM STUDY

2.1 FEASIBILITY STUDY

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

Three key considerations involved in the feasibility analysis are

- ◆ **ECONOMICAL FEASIBILITY**

- ◆ **TECHNICAL FEASIBILITY**

- ◆ **SOCIAL FEASIBILITY**

- **ECONOMICAL FEASIBILITY**

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

- **TECHNICAL FEASIBILITY**

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

➤ **SOCIAL FEASIBILITY**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

Disadvantages

- In the existing work, the system doesn't have more security due to lack of Delegation-aware encryption and Adjustable onion encryption.
- The system proposes a key assignment scheme to simplify key management in hierarchical access control policy. Also, this work does not consider policy update issues..

2.2 PROPOSED SYSTEM

- ❖ The proposed system presents Crypt-DAC, a cryptographically enforced dynamic access control system on untrusted cloud. Crypt-DAC delegates the cloud to update encrypted files in permission revocations. In Crypt-DAC, a file is encrypted by a symmetric key list which records a file key and a sequence of revocation keys. In a revocation, the administrator uploads a new revocation key to the cloud, which encrypts the file with a new layer of encryption and updates the encrypted key list accordingly. Same as previous works, we assume a honest-but-curious cloud, i.e., the cloud is honest to perform the required commands (such as re-encryption of files and properly update previous encrypted **files**) but is curious to passively gathering sensitive information. Although the basic idea of layered encryption is simple, it entails tremendous technical challenges. For instance, the size of key list and encryption layers would increase as the number of revocation operations, which incurs additional decryption overhead for users to access files. To overcome such a problem, Crypt-DAC proposes three key techniques as follows.
- ❖ First, Crypt-DAC proposes delegation-aware encryption strategy to delegate the cloud to update policy data. For a file, the administrator appends a new revocation key at the end of its key list and requests the cloud to update this key list in the policy data. The size of the key list however increases with the revocation operations, and a user has to download and decrypt a large key list in each file access. To overcome this problem, we adopt the key rotation technique to compactly encrypt the key list in the policy data. As a result, the size of the key list remains constant regardless of revocation operations.

- ❖ Second, Crypt-DAC proposes adjustable onion encryption strategy to delegate the cloud to update file data. For a file, the administrator requests the cloud to encrypt the file with a new layer of encryption. Similarly, the size of the encryption layers increases with the revocation operations, and a user has to decrypt multiple times in each file access. To overcome this problem, we enable the administrator to define a tolerable bound for the file. Once the size of encryption layers reaches the bound, it can be made to not increase anymore by delegating encryption operations to the cloud. As a result, the administrator can flexibly adjust a tolerable bound for each file (according to file type, access pattern, etc.) to achieve a balance between efficiency and security.
- ❖ Crypt- DAC proposes delayed de-onion encryption strategy to periodically refresh the symmetric key list of the file and remove the bounded encryption layers over it through writing operations. In specific, the next user to write to the file encrypts the writing content by a new symmetric key list only containing a new file key, and updates the key list in the policy data. With this strategy, Crypt-DAC periodically removes the bounded encryption layers of files while amortizing the burden to a large number of writing users.

Advantages

- Crypt-DAC achieves efficient revocation, efficient file access and immediate revocation simultaneously.
- The system stores encrypted data on the cloud, but never reveals the decryption keys to the cloud. This protects the confidentiality of the file data

3. SYSTEM REQUIREMENTS

- Operating System - Windows XP
- Coding Language - Java/J2EE(JSP,Servlet)
- Front End - J2EE
- Back End - MySQL

HARDWARE REQUIREMENTS:-

H/W System Configuration:-

- Processor - Pentium –IV
- RAM - 4 GB (min)
- Hard Disk - 20 GB
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
- Monitor - SVGA

4. TECHNOLOGICAL DESCRIPTION

4.1 Client Server

Over view:

With the varied topic in existence in the fields of computers, Client Server is one, which has generated more heat than light, and also more hype than reality. This technology has acquired a certain critical mass attention with its dedication conferences and magazines. Major computer vendors such as IBM and DEC, have declared that Client Servers is their main future market. A survey of DBMS magazine revealed that 76% of its readers were actively looking at the client server solution. The growth in the client server development tools from \$200 million in 1992 to more than \$1.2 billion in 1996.

Client server implementations are complex but the underlying concept is simple and powerful. A client is an application running with local resources but able to request the database and relate the services from separate remote server. The software mediating this client server interaction is often referred to as MIDDLEWARE.

The typical client either a PC or a Work Station connected through a network to a more powerful PC, Workstation, Midrange or Main Frames server usually capable of handling request from more than one client. However, with some configuration server may also act as client. A server may need to access other server in order to process the original client request.

The key client server idea is that client as user is essentially insulated from the physical location and formats of the data needs for their application. With the proper middleware, a client input from or report can transparently access and manipulate both local database on the client machine and remote databases on one or more servers. An added bonus is the client server opens the door to multi-vendor database access indulging heterogeneous table joins.

What is a Client Server

Two prominent systems in existence are client server and file server systems. It is essential to distinguish between client servers and file server systems. Both provide shared network access to data but the comparison ends there! The file server simply provides a remote disk drive that can be accessed by LAN applications on a file by file basis. The client server offers full relational database services such as SQL-Access, Record modifying, Insert, Delete with full relational integrity backup/ restore performance for high volume of transactions, etc. the

client server middleware provides a flexible interface between client and server, who does what, when and to whom.

Why Client Server

Client server has evolved to solve a problem that has been around since the earliest days of computing: how best to distribute your computing, data generation and data storage resources in order to obtain efficient, cost effective departmental and enterprise wide data processing. During mainframe era choices were quite limited. A central machine housed both the CPU and DATA (cards, tapes, drums and later disks). Access to these resources was initially confined to batched runs that produced departmental reports at the appropriate intervals. A strong central information service department ruled the corporation. The role of the rest of the corporation limited to requesting new or more frequent reports and to provide hand written forms from which the central data banks were created and updated. The earliest client server solutions therefore could best be characterized as “SLAVE-MASTER”.

Time-sharing changed the picture. Remote terminal could view and even change the central data, subject to access permissions.

Front end or User Interface Design

The entire user interface is planned to be developed in browser specific environment with a touch of Intranet-Based Architecture for achieving the Distributed Concept. The browser specific components are designed by using the HTML standards, and the dynamism of the designed by concentrating on the constructs of the Java Server Pages.

Communication or Database Connectivity Tier

The Communication architecture is designed by concentrating on the Standards of Servlets and Enterprise Java Beans. The database connectivity is established by using the Java Data Base Connectivity. The standards of three-tier architecture are given major concentration to keep the standards of higher cohesion and limited coupling for effectiveness of the operations.

4.2 Features of The Language Used

In my project, I have chosen *Java* language for developing the code.

About Java

Initially the language was called as “oak” but it was renamed as “Java” in 1995. The primary motivation of this language was the need for a platform-independent (i.e., architecture neutral) language that could be used to create software to be embedded in various consumer electronic devices.

- Java is a programmer’s language.
- Java is cohesive and consistent.
- Except for those constraints imposed by the Internet environment, Java gives the programmer, full control.

Finally, Java is to Internet programming where C was to system programming.

Importance of Java to the Internet

Java has had a profound effect on the Internet. This is because; Java expands the Universe of objects that can move about freely in Cyberspace. In a network, two categories of objects are transmitted between the Server and the Personal computer. They are: Passive information and Dynamic active programs. The Dynamic, Self-executing programs cause serious problems in the areas of Security and probability. But, Java addresses those concerns and by doing so, has opened the door to an exciting new form of program called the Applet.

Java can be used to create two types of programs

Applications and Applets: An application is a program that runs on our Computer under the operating system of that computer. It is more or less like one creating using C or C++. Java’s ability to create Applets makes it important. An Applet is an application designed to be transmitted over the Internet and executed by a Java –compatible web browser. An applet is actually a tiny Java program, dynamically downloaded across the network, just like an image. But the difference is, it is an intelligent program, not just a media file. It can react to the user input and dynamically change.

Features Of Java

Security

Every time you that you download a “normal” program, you are risking a viral infection. Prior to Java, most users did not download executable programs frequently, and those who did scanned them for viruses prior to execution. Most users still worried about the possibility of infecting their systems with a virus. In addition, another type of malicious program exists that must be guarded against. This type of program can gather private information, such as credit card numbers, bank account balances, and passwords. Java answers both these concerns by providing a “firewall” between a network application and your computer.

When you use a Java-compatible Web browser, you can safely download Java applets without fear of virus infection or malicious intent.

Portability

For programs to be dynamically downloaded to all the various types of platforms connected to the Internet, some means of generating portable executable code is needed .As you will see, the same mechanism that helps ensure security also helps create portability. Indeed, Java’s solution to these two problems is both elegant and efficient.

The Byte code

The key that allows the Java to solve the security and portability problems is that the output of Java compiler is Byte code. Byte code is a highly optimized set of instructions designed to be executed by the Java run-time system, which is called the Java Virtual Machine (JVM). That is, in its standard form, the JVM is an interpreter for byte code.

Translating a Java program into byte code helps makes it much easier to

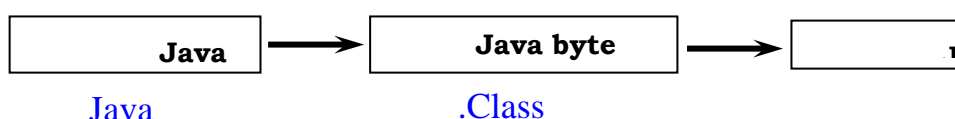
run a program in a wide variety of environments. The reason is, once the run-time package exists for a given system, any Java program can run on it.

Although Java was designed for interpretation, there is technically nothing about Java that prevents on-the-fly compilation of byte code into native code. Sun has just completed its Just In Time (JIT) compiler for byte code. When the JIT compiler is a part of JVM, it compiles byte code into executable code in real time, on a piece-by-piece, demand basis. It is not possible to compile an entire Java program into executable code all at once, because Java performs various run-time checks that can be done only at run time. The JIT compiles code, as it is needed, during execution.

Java, Virtual Machine (JVM)

Beyond the language, there is the Java virtual machine. The Java virtual machine is an important element of the Java technology. The virtual machine can be embedded within a web browser or an operating system. Once a piece of Java code is loaded onto a machine, it is verified. As part of the loading process, a class loader is invoked and does byte code verification makes sure that the code that's has been generated by the compiler will not corrupt the machine that it's loaded on. Byte code verification takes place at the end of the compilation process to make sure that is all accurate and correct. So byte code verification is integral to the compiling and executing of Java code.

Overall Description



Picture showing the development process of JAVA Program

Java programming uses to produce byte codes and executes them. The first box indicates that the Java source code is located in a Java file that is processed with a Java compiler called javac. The Java compiler produces a file called a. class file, which contains the byte code. The. Class file is then loaded across the network or loaded locally on your machine into the execution environment is the Java virtual machine, which interprets and executes the byte code.

Java Architecture

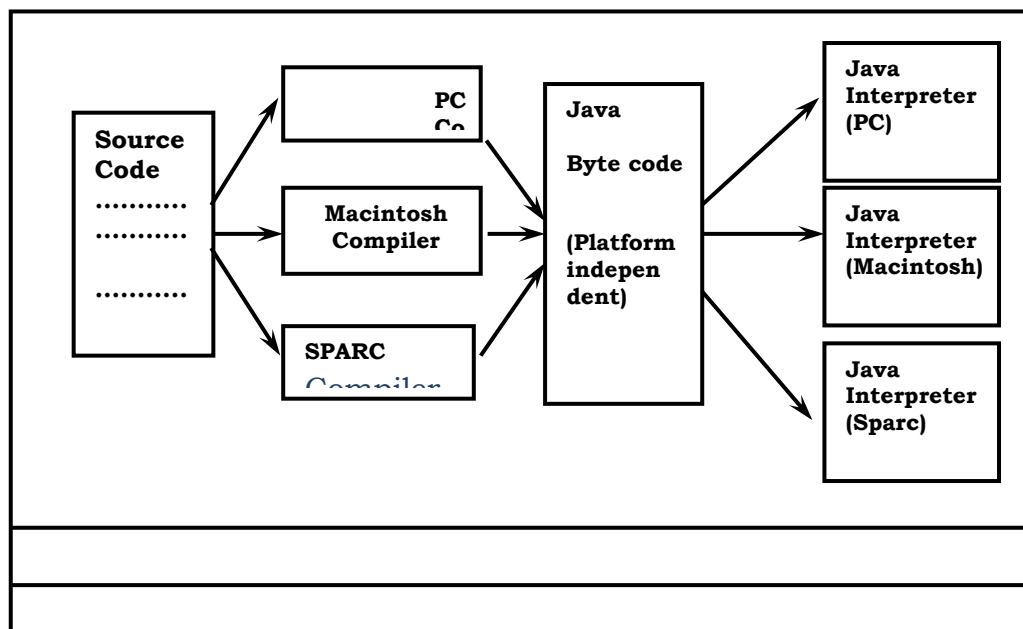
Java architecture provides a portable, robust, high performing

environment for development. Java provides portability by compiling the byte codes for the Java Virtual Machine, which is then interpreted on each platform by the run-time environment. Java is a dynamic system, able to load code when needed from a machine in the same room or across the planet.

Compilation of code

When you compile the code, the Java compiler creates machine code (called byte code) for a hypothetical machine called Java Virtual Machine (JVM). The JVM is supposed to execute the byte code. The JVM is created for overcoming the issue of portability. The code is written and compiled for one machine and interpreted on all machines. This machine is called Java Virtual Machine.

FIG: Compiling and interpreting Java Source Code



During run-time the Java interpreter tricks the byte code file into thinking that it is running on a Java Virtual Machine. In reality this could be a Intel Pentium Windows 95 or Sun SARC station running Solaris or Apple Macintosh running system and all could receive code from any computer through Internet and run the Applets.

Simple

Java was designed to be easy for the Professional programmer to learn and to use effectively. If you are an experienced C++ programmer, learning Java will be even easier. Because Java inherits the C/C++ syntax and many of the object oriented features of C++. Most of the confusing concepts from C++ are either left out of Java or implemented in a cleaner, more approachable manner. In Java there are a small

number of clearly defined ways to accomplish a given task.

Object-Oriented

Java was not designed to be source-code compatible with any other language. This allowed the Java team the freedom to design with a blank slate. One outcome of this was a clean usable, pragmatic approach to objects. The object model in Java is simple and easy to extend, while simple types, such as integers, are kept as high-performance non-objects.

Robust

The multi-platform environment of the Web places extraordinary demands on a program, because the program must execute reliably in a variety of systems. The ability to create robust programs was given a high priority in the design of Java. Java is strictly typed language; it checks your code at compile time and run time.

Java virtually eliminates the problems of memory management and deallocation, which is completely automatic. In a well-written Java program, all run time errors can –and should –be managed by your program.

4.3 JDBC connectivity

The JDBC provides database-independent connectivity between the J2EE platform and a wide range of tabular data sources. JDBC technology allows an Application Component Provider to:

- Perform connection and authentication to a database server
- Manager transactions
- Move SQL statements to a database engine for preprocessing and execution
- Execute stored procedures
- Inspect and modify the results from Select statements.

4.4 Tomcat 6.0 web server

Tomcat is an open source web server developed by Apache Group. Apache Tomcat is the servlet container that is used in the official Reference Implementation for the Java Servlet and Java Server Pages technologies. The Java Servlet and Java Server Pages specifications are developed by Sun under the Java Community Process. Web Servers like Apache Tomcat support only web components while an application server supports web components as well as business components (BEAs Weblogic, is one of the popular application server). To develop a web application with jsp/servlet install any web server like JRun, Tomcat etc to run your application.



5. SYSTEM DESIGN

System design is the phase that bridges the gap between problem domain and the existing system in a manageable way. This phase focuses on the solution domain.

In this phase, the complex activity of system development is divided into several smaller sub-activities, which coordinate with each other to achieve the main objective of system development.

There are Three main user and Admin

- 1.DATA OWNE
2. DATAUSER
- 3.CLOUD SERVER
4. ADMIN

1. WORK OF DATA OWNER

- View My Profile
- Upload,View My Files
- View Search Control & Dec Key Permitted Files
- Verify,Delete File
- Select File Name
- Owner Name
- Owner File

2. WORK OF DATA USER

- View My Profile
- Send Search Request
- Search File
- View Cloud Files
- Request secret Key
- Request DAC
- View AccessControl Responses
- Download fileName
- Rank
- Username
- secret_key

3. CLOUD SERVER

- View Owners & Authorize
- View Users & Authorize
- View Cloud Server Files
- View Data User Key Access Control Request

- View Search Access Control Request
- View Access Policy Details and Its Update
- View Transactions
- View Attackers
- View File's Rank in Chart
- View Time Delay in Chart
- View Throughput in Chart
- File ID
- File Name
- Owner Name
- Secret Key
- User Details
- Trapdoor
- Exit

4. ADMIN

- View DAC Request and Permit
- View Data Transactions
- File name dac

➤ Inputs to System Design

System design takes the following inputs –

- Statement of work
- Requirement determination plan
- Current situation analysis
- Proposed system requirements including a conceptual data model, modified DFDs, and Metadata (data about data).

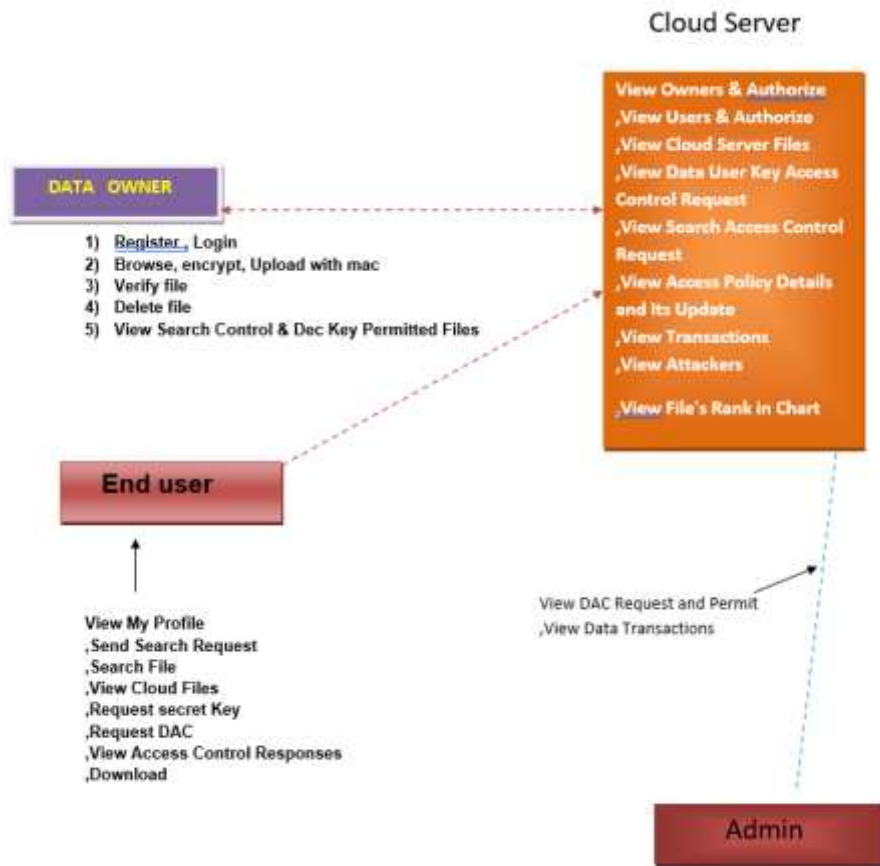
➤ Outputs for System Design

System design gives the following outputs –

- Infrastructure and organizational changes for the proposed system.
- A data schema, often a relational schema.
- Metadata to define the tables/files and columns/data-items.
- A function hierarchy diagram or web page map that graphically describes the program structure.
- Actual or pseudocode for each module in the program.
- A prototype for the proposed system.

ARCHITECTURE

1. Architecture diagram



Each Cloud server

1. View all users (data owners & End Users)
2. View all files (fname,owner name,RSA secret key,MAC,UploadDateTime,FileContents)
3. View all attackers
4. View end user service request (ie file request)
5. Process all file requests
6. Recover attacked service and then process.

Table for Service Queue

<u>End username</u>	<u>File Request</u>	<u>Queue Priority</u>	<u>Req Date Time</u>	<u>Processed Status</u>	<u>Attacked Status</u>	<u>Recover</u>	<u>Process</u>
Manju	<u>Test.java</u>	<u>1</u>	<u>17/10/2015:5:27</u>	<u>Waiting</u>	<u>No</u>	<u>Yes</u>	<u>Response Service</u>
Harish	<u>Test1.java</u>	<u>2</u>	<u>17/10/2015:8:27</u>	<u>Processed</u>	<u>No</u>	<u>Yes</u>	<u>Service Processed</u>

2. Data Flow CHART :

The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of the input data to the system, various processing carried out on these data, and the output data is generated by the system.

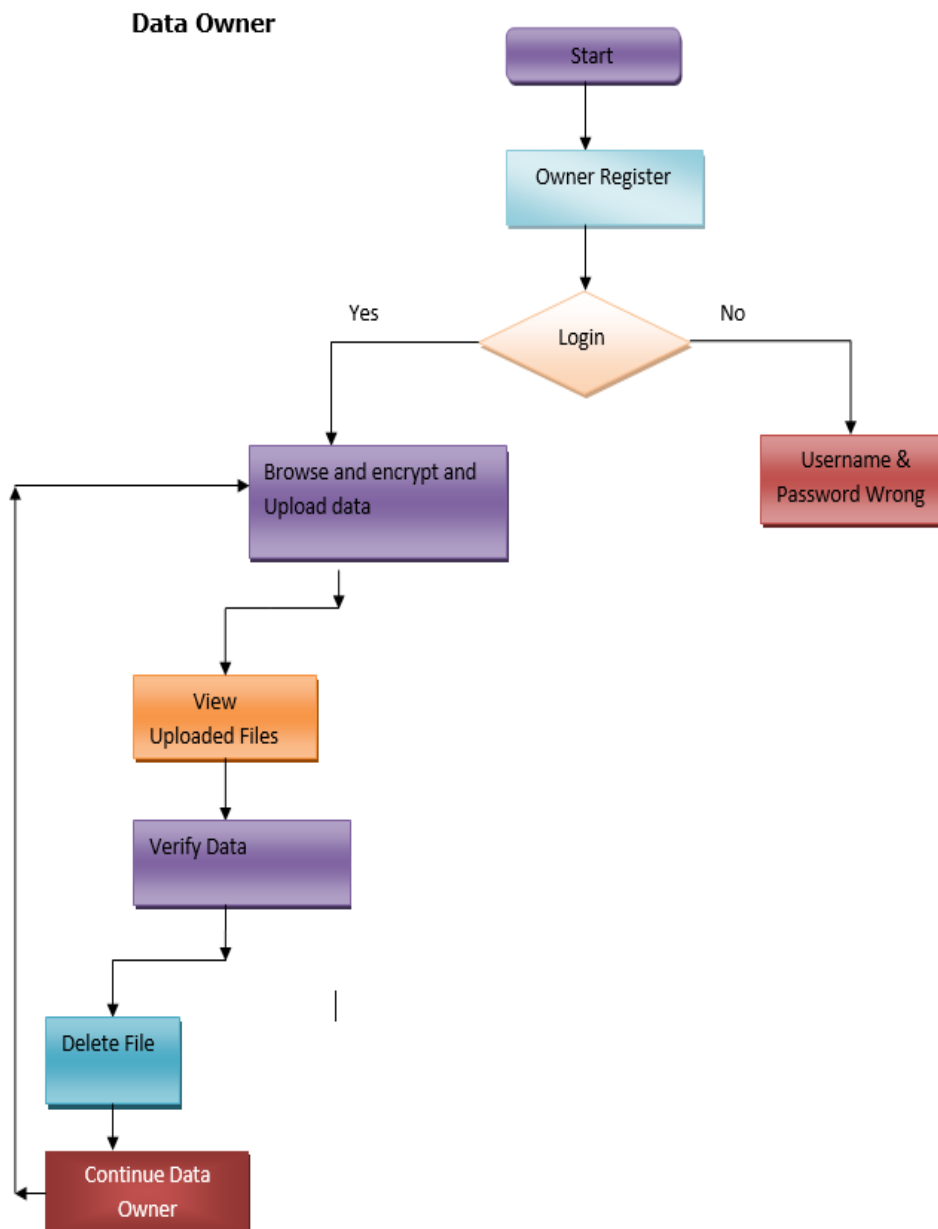


Fig:5.1 Data flow chart

3. DATA USER DIAGRAM

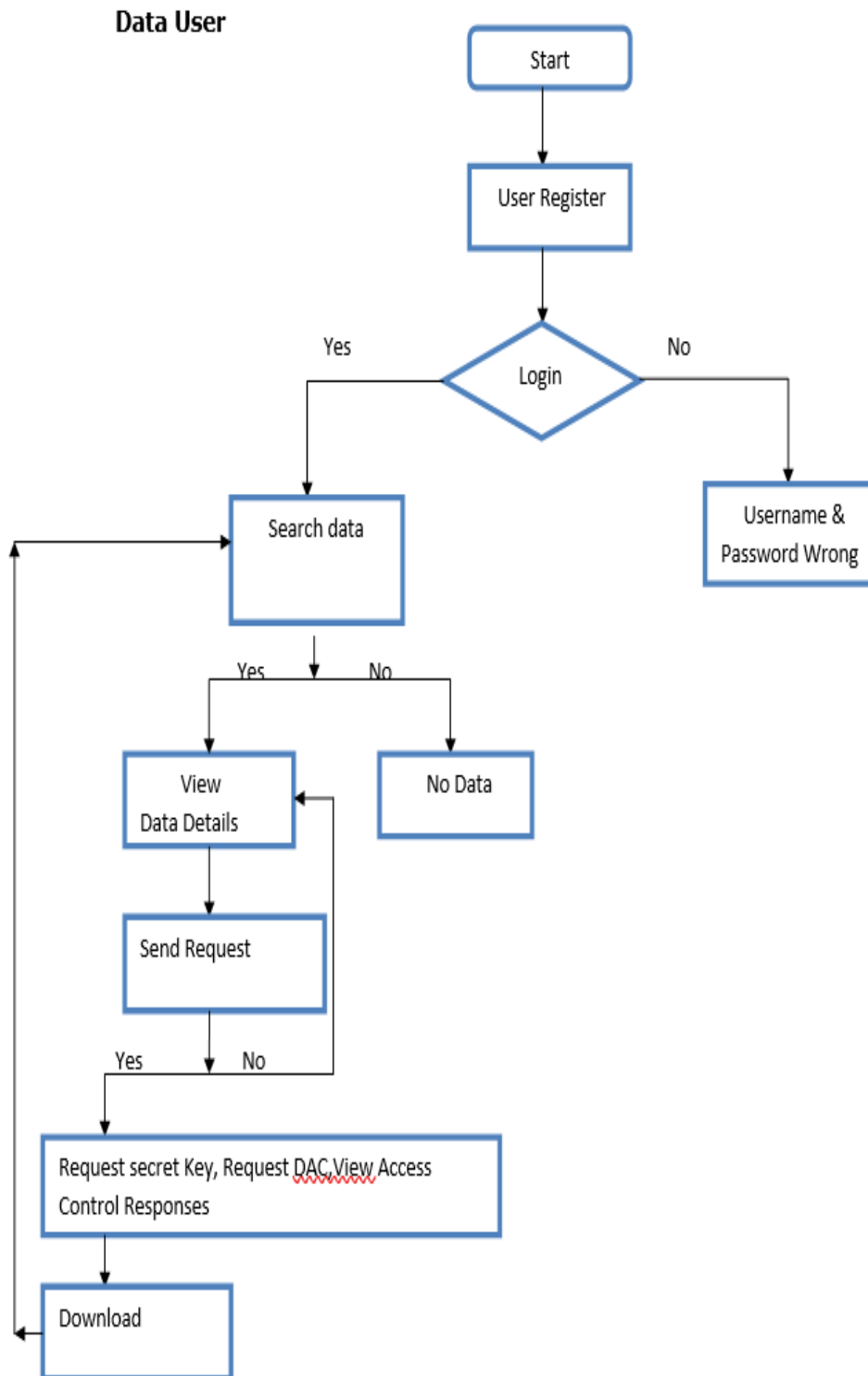


Fig:5.2 DATA USER DIAGRAM

4. Cloud server Flow Diagram

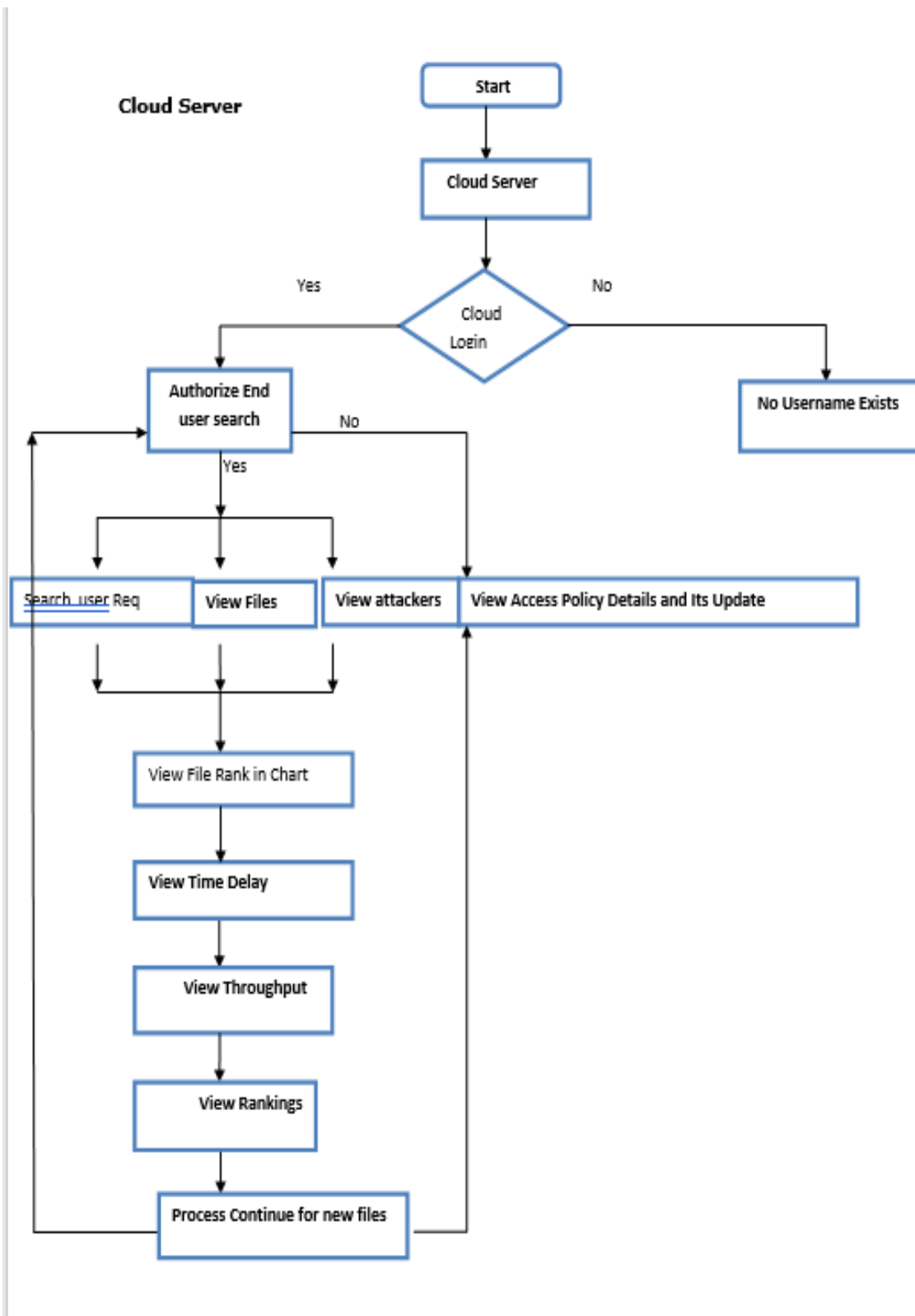
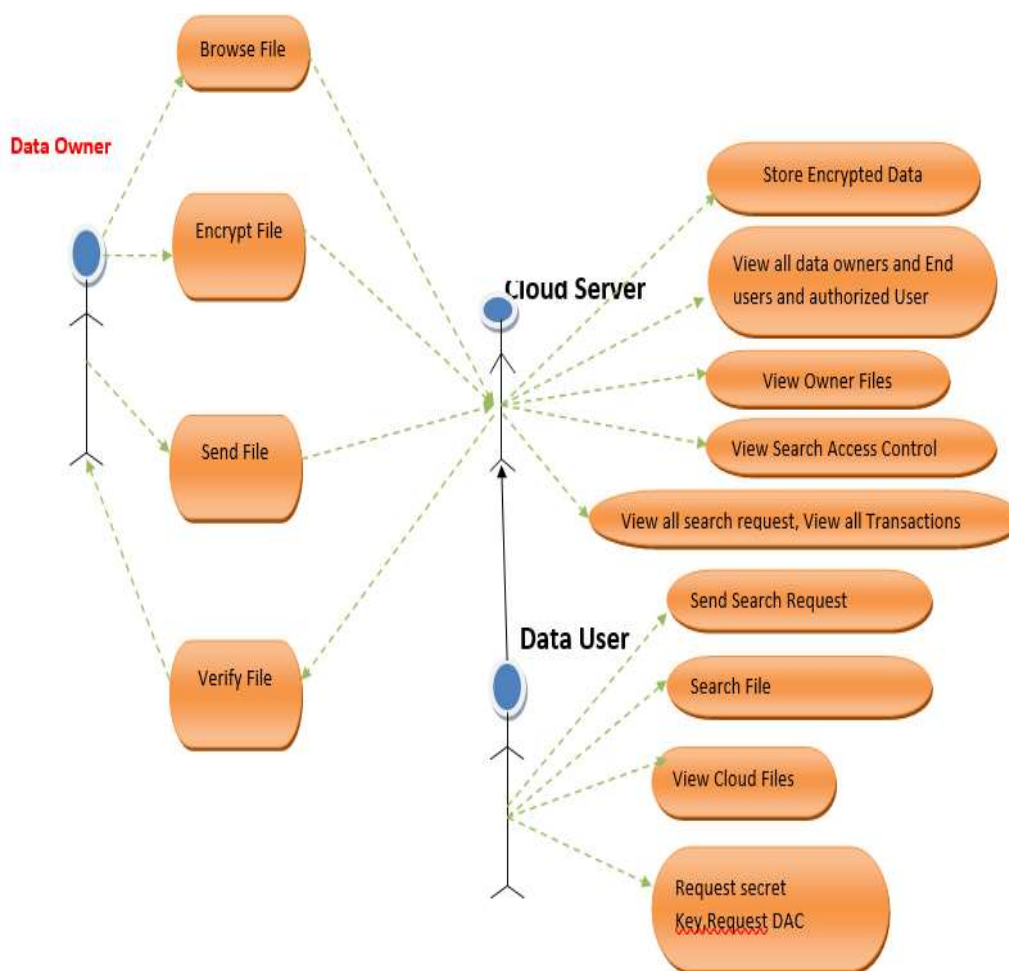


Fig:5.3 Cloud Flow Chart

USE CASE DIAGRAM:

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



CLASS DIAGRAM:

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

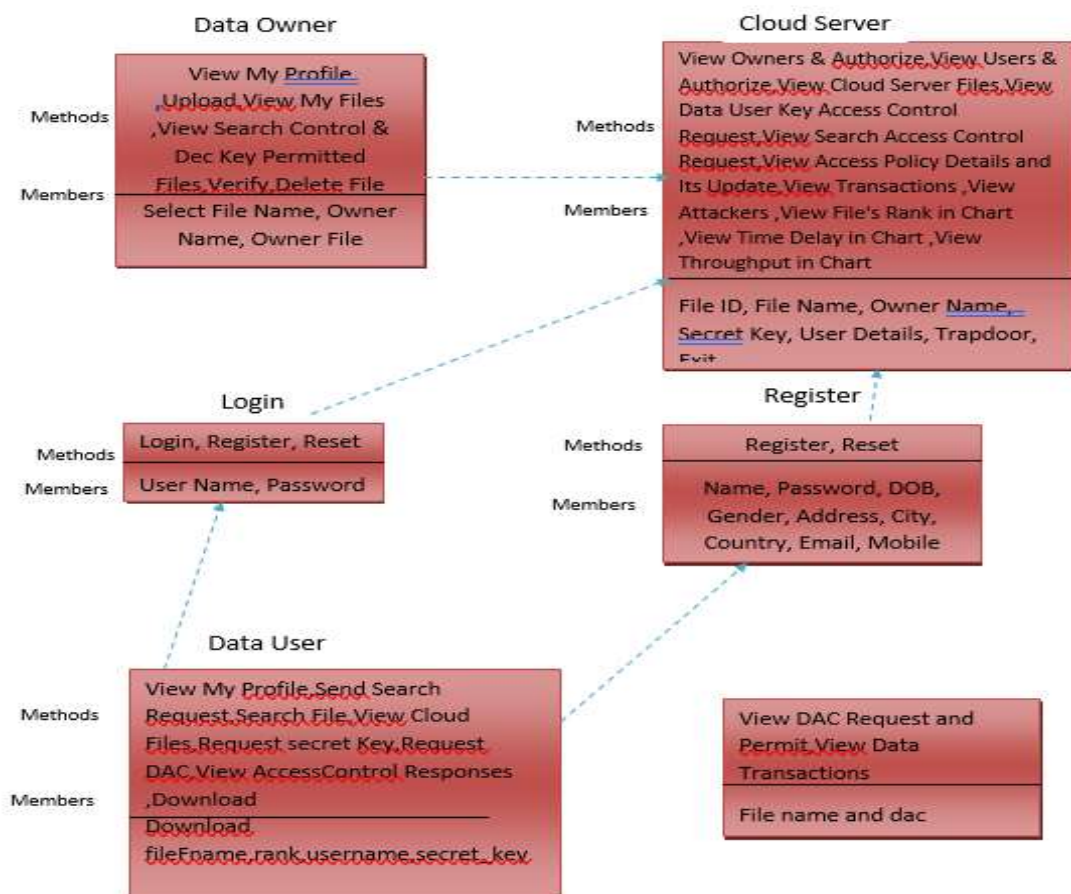


Fig:5.3 Class Diagram

SEQUENCE DIAGRAM:

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

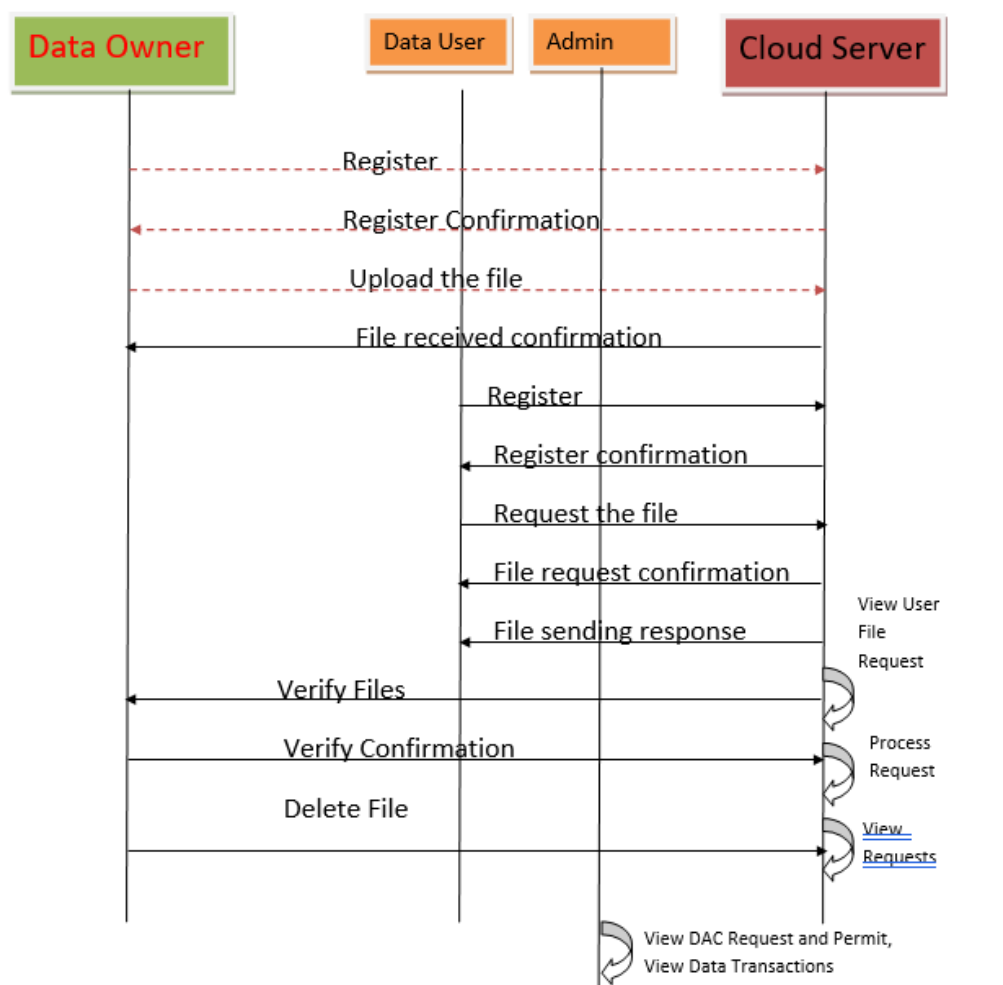


Fig:5.4 Sequence diagram

6. IMPLEMENTATION

A. MODULER DESCRIPTION

1. DATA OWNER
2. CLOUD SERVER
3. END USER

1. Data Owner

In this module, the data owner uploads their data with its chunks in the cloud server. For the security purpose the data owner encrypts the data file's chunks and then store in the cloud. The data owner can change the policy over data files by updating the expiration time. The Data owner can have capable of manipulating the encrypted data file. And the data owner can set the access privilege to the encrypted data file.

Dynamic Operation

Upload: is the operation to encrypt and upload the file

Delete: Is the operation to delete a corresponding data owner file in the cloud.

Verify: Verifying the data whether it is safe or not in the cloud.

2. Cloud Server

The cloud service provider manages a cloud to provide data storage service. Data owners encrypt their data files and store them in the cloud for sharing with data consumers. To access the shared data files, data consumers download encrypted data files of their interest from the cloud and then decrypt them. The end user request will be processes based on the queue.

3. End User

The Cloud User who has a large amount of data to be stored in multiple clouds and have the permissions to access and manipulate stored data. The end user sends the request for corresponding file request and it will be processed in the cloud based on the queue and response to the end user.

B. Java Technology

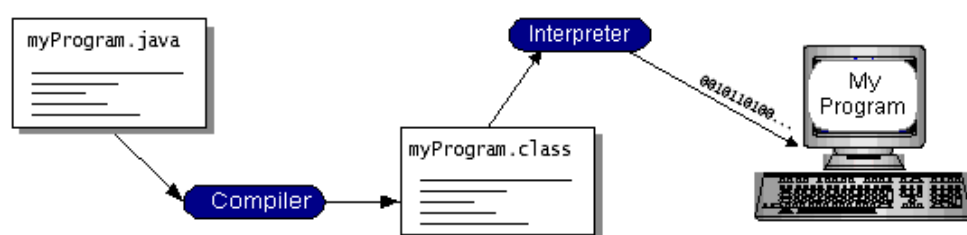
Java technology is both a programming language and a platform.

The Java Programming Language

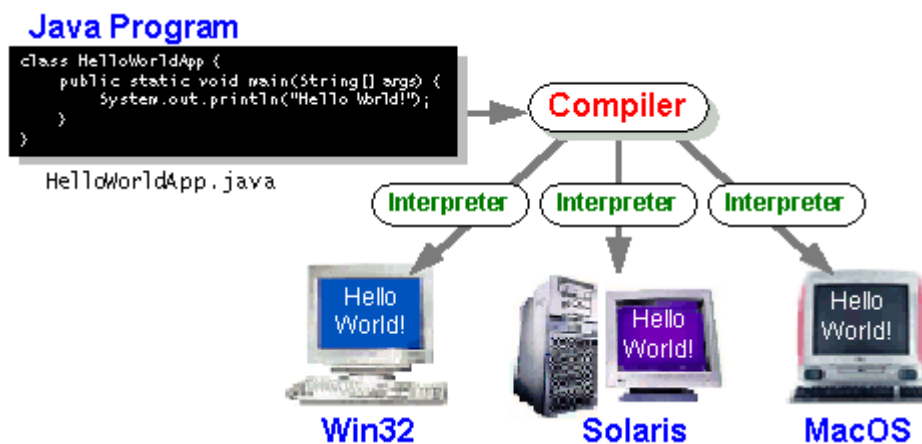
The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

- Simple
- Architecture neutral
- Object oriented
- Portable
- Distributed
- High performance
- Interpreted
- Multithreaded
- Robust
- Dynamic
- Secure

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called *Java byte codes*—the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The following figure illustrates how this works.



You can think of Java byte codes as the machine code instructions for the *Java Virtual Machine* (Java VM). Every Java interpreter, whether it's a development tool or a Web browser that can run applets, is an implementation of the Java VM. Java byte codes help make “write once, run anywhere” possible. You can compile your program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the Java VM. That means that as long as a computer has a Java VM, the same program written in the Java programming language can run on Windows 2000, a Solaris workstation, or on an iMac.



The Java Platform

A *platform* is the hardware or software environment in which a program runs. We've already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and MacOS. Most platforms can be described as a combination of the operating system and hardware. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms.

The Java platform has two components:

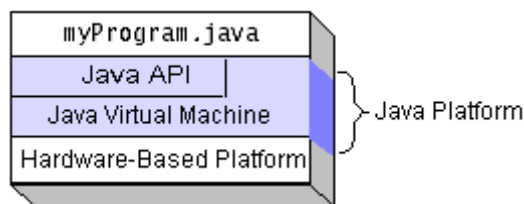
- The *Java Virtual Machine* (Java VM)
- The *Java Application Programming Interface* (Java API)

You've already been introduced to the Java VM. It's the base for the Java platform and is ported onto various hardware-based platforms.

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI)

widgets. The Java API is grouped into libraries of related classes and interfaces; these libraries are known as *packages*. The next section, What Can Java Technology Do? Highlights what functionality some of the packages in the Java API provide.

The following figure depicts a program that's running on the Java platform. As the figure shows, the Java API and the virtual machine insulate the program from the hardware.



Native code is code that after you compile it, the compiled code runs on a specific hardware platform. As a platform-independent environment, the Java platform can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just-in-time byte code compilers can bring performance close to that of native code without threatening portability.

What Can Java Technology Do?

The most common types of programs written in the Java programming language are *applets* and *applications*. If you've surfed the Web, you're probably already familiar with applets. An applet is a program that adheres to certain conventions that allow it to run within a Java-enabled browser.

However, the Java programming language is not just for writing cute, entertaining applets for the Web. The general-purpose, high-level Java programming language is also a powerful software platform. Using the generous API, you can write many types of programs.

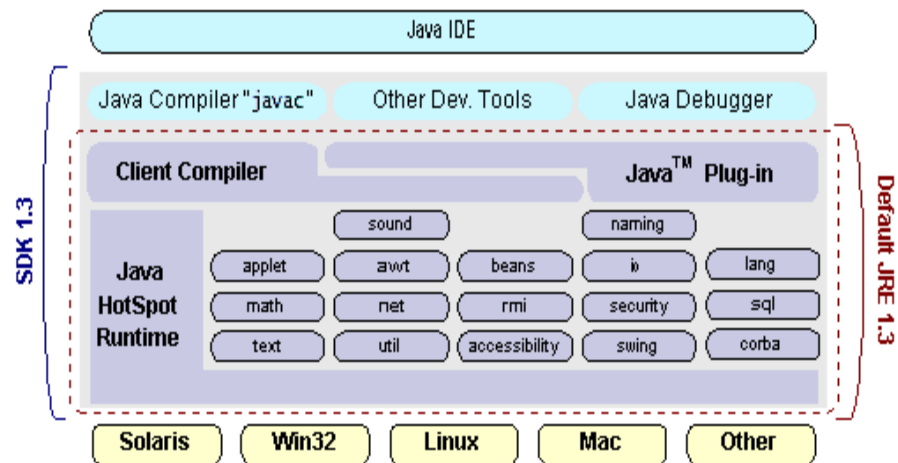
An application is a standalone program that runs directly on the Java platform. A special kind of application known as a *server* serves and supports clients on a network. Examples of servers are Web servers, proxy servers, mail servers, and print servers. Another specialized program is a *servlet*. A servlet can almost be thought of as an applet that runs on the server side. Java Servlets are a popular choice for building interactive web applications, replacing the use of CGI scripts. Servlets are similar to applets

in that they are runtime extensions of applications. Instead of working in browsers, though, servlets run within Java Web servers, configuring or tailoring the server.

How does the API support all these kinds of programs? It does so with packages of software components that provides a wide range of functionality. Every full implementation of the Java platform gives you the following features:

- **The essentials:** Objects, strings, threads, numbers, input and output, data structures, system properties, date and time, and so on.
- **Applets:** The set of conventions used by applets.
- **Networking:** URLs, TCP (Transmission Control Protocol), UDP (User Data gram Protocol) sockets, and IP (Internet Protocol) addresses.
- **Internationalization:** Help for writing programs that can be localized for users worldwide. Programs can automatically adapt to specific locales and be displayed in the appropriate language.
- **Security:** Both low level and high level, including electronic signatures, public and private key management, access control, and certificates.
- **Software components:** Known as JavaBeansTM, can plug into existing component architectures.
- **Object serialization:** Allows lightweight persistence and communication via Remote Method Invocation (RMI).
- **Java Database Connectivity (JDBCTM):** Provides uniform access to a wide range of relational databases.

The Java platform also has APIs for 2D and 3D graphics, accessibility, servers, collaboration, telephony, speech, animation, and more. The following figure depicts what is included in the Java 2 SDK.



How Will Java Technology Change My Life?

We can't promise you fame, fortune, or even a job if you learn the Java programming language. Still, it is likely to make your programs better and requires less effort than other languages. We believe that Java technology will help you do the following:

- **Get started quickly:** Although the Java programming language is a powerful object-oriented language, it's easy to learn, especially for programmers already familiar with C or C++.
- **Write less code:** Comparisons of program metrics (class counts, method counts, and so on) suggest that a program written in the Java programming language can be four times smaller than the same program in C++.
- **Write better code:** The Java programming language encourages good coding practices, and its garbage collection helps you avoid memory leaks. Its object orientation, its JavaBeans component architecture, and its wide-ranging, easily extendible API let you reuse other people's tested code and introduce fewer bugs.
- **Develop programs more quickly:** Your development time may be as much as twice as fast versus writing the same program in C++. Why? You write fewer lines of code and it is a simpler programming language than C++.
- **Avoid platform dependencies with 100% Pure Java:** You can keep your program portable by avoiding the use of libraries written in other languages. The 100% Pure Java™ Product Certification

Program has a repository of historical process manuals, white papers, brochures, and similar materials online.

- **Write once, run anywhere:** Because 100% Pure Java programs are compiled into machine-independent byte codes, they run consistently on any Java platform.
- **Distribute software more easily:** You can upgrade applets easily from a central server. Applets take advantage of the feature of allowing new classes to be loaded “on the fly,” without recompiling the entire program.

ODBC

Microsoft Open Database Connectivity (ODBC) is a standard programming interface for application developers and database systems providers. Before ODBC became a *de facto* standard for Windows programs to interface with database systems, programmers had to use proprietary languages for each database they wanted to connect to. Now, ODBC has made the choice of the database system almost irrelevant from a coding perspective, which is as it should be. Application developers have much more important things to worry about than the syntax that is needed to port their program from one database to another when business needs suddenly change.

Through the ODBC Administrator in Control Panel, you can specify the particular database that is associated with a data source that an ODBC application program is written to use. Think of an ODBC data source as a door with a name on it. Each door will lead you to a particular database. For example, the data source named Sales Figures might be a SQL Server database, whereas the Accounts Payable data source could refer to an Access database. The physical database referred to by a data source can reside anywhere on the LAN.

The ODBC system files are not installed on your system by Windows 95. Rather, they are installed when you setup a separate database application, such as SQL Server Client or Visual Basic 4.0. When the ODBC icon is installed in Control Panel, it uses a file called ODBCINST.DLL. It is also possible to administer your ODBC data sources through a stand-alone program called ODBCADM.EXE. There is a 16-bit and a 32-bit version of this program and each maintains a separate list of ODBC data sources.

From a programming perspective, the beauty of ODBC is that the application can be written to use the same set of function calls to interface with any data source, regardless of the database vendor. The source code of the application doesn't change whether it talks to Oracle or SQL Server. We only mention these two as an example. There are ODBC drivers available for several dozen popular database systems. Even Excel spreadsheets and plain text files can be turned into data sources. The operating system uses the Registry information written by ODBC Administrator to determine which low-level ODBC drivers are needed to talk to the data source (such as the interface to Oracle or SQL Server). The loading of the ODBC drivers is transparent to the ODBC application program. In a client/server environment, the ODBC API even handles many of the network issues for the application programmer.

The advantages of this scheme are so numerous that you are probably thinking there must be some catch. The only disadvantage of ODBC is that it isn't as efficient as talking directly to the native database interface. ODBC has had many detractors make the charge that it is too slow. Microsoft has always claimed that the critical factor in performance is the quality of the driver software that is used. In our humble opinion, this is true. The availability of good ODBC drivers has improved a great deal recently. And anyway, the criticism about performance is somewhat analogous to those who said that compilers would never match the speed of pure assembly language. Maybe not, but the compiler (or ODBC) gives you the opportunity to write cleaner programs, which means you finish sooner. Meanwhile, computers get faster every year.

JDBC

In an effort to set an independent database standard API for Java; Sun Microsystems developed Java Database Connectivity, or JDBC. JDBC offers a generic SQL database access mechanism that provides a consistent interface to a variety of RDBMSs. This consistent interface is achieved through the use of "plug-in" database connectivity modules, or *drivers*. If a database vendor wishes to have JDBC support, he or she must provide the driver for each platform that the database and Java run on.

To gain a wider acceptance of JDBC, Sun based JDBC's framework on ODBC. As you discovered earlier in this chapter, ODBC has widespread support on a variety of platforms. Basing JDBC on ODBC will allow vendors to bring JDBC drivers to market much faster than developing a completely new connectivity solution.

JDBC was announced in March of 1996. It was released for a 90 day public review that ended June 8, 1996. Because of user input, the final JDBC v1.0 specification was released soon after.

JDBC Goals

Few software packages are designed without goals in mind. JDBC is one that, because of its many goals, drove the development of the API. These goals, in conjunction with early reviewer feedback, have finalized the JDBC class library into a solid framework for building database applications in Java.

The goals that were set for JDBC are important. They will give you some insight as to why certain classes and functionalities behave the way they do. The eight design goals for JDBC are as follows:

1. *SQL Level API*

The designers felt that their main goal was to define a SQL interface for Java. Although not the lowest database interface level possible, it is at a low enough level for higher-level tools and APIs to be created. Conversely, it is at a high enough level for application programmers to use it confidently. Attaining this goal allows for future tool vendors to “generate” JDBC code and to hide many of JDBC's complexities from the end user.

2. *SQL Conformance*

SQL syntax varies as you move from database vendor to database vendor. In an effort to support a wide variety of vendors, JDBC will allow any query statement to be passed through it to the underlying database driver. This allows the connectivity module to handle non-standard functionality in a manner that is suitable for its users.

3. ***JDBC must be implemental on top of common database interfaces***

The JDBC SQL API must “sit” on top of other common SQL level APIs. This goal allows JDBC to use existing ODBC level drivers by the use of a software interface. This interface would translate JDBC calls to ODBC and vice versa.

4. ***Provide a Java interface that is consistent with the rest of the Java system***

Because of Java’s acceptance in the user community thus far, the designers feel that they should not stray from the current design of the core Java system.

5. ***Keep it simple***

This goal probably appears in all software design goal listings. JDBC is no exception. Sun felt that the design of JDBC should be very simple, allowing for only one method of completing a task per mechanism. Allowing duplicate functionality only serves to confuse the users of the API.

6. ***Use strong, static typing wherever possible***

Strong typing allows for more error checking to be done at compile time; also, less error appear at runtime.

1. ***Keep the common cases simple***

Because more often than not, the usual SQL calls used by the programmer are simple SELECT’s, INSERT’s, DELETE’s and UPDATE’s, these queries should be simple to perform with JDBC. However, more complex SQL statements should also be possible.

Finally we decided to proceed the implementation using Java Networking.

And for dynamically updating the cache table we go for MS Access database.

Java ha two things: a programming language and a platform.

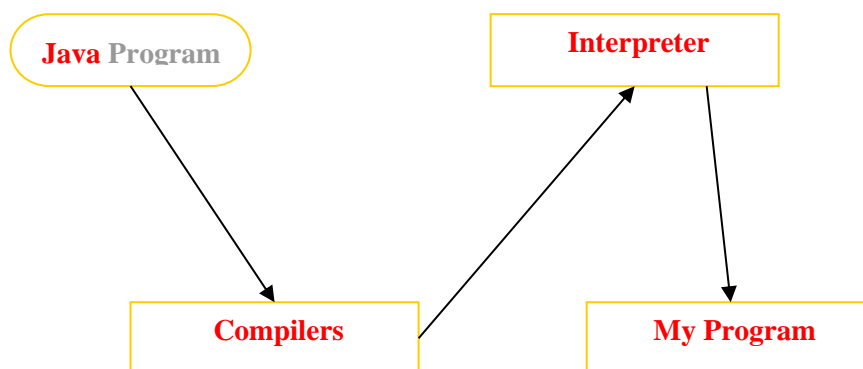
Java is a high-level programming language that is all of the following

Simple	Architecture-	neutral
Object-oriented	Portable	
Distributed	High-performance	

Interpreted	multithreaded
Robust	Dynamic
Secure	

Java is also unusual in that each Java program is both compiled and interpreted. With a compiler you translate a Java program into an intermediate language called Java byte codes the platform-independent code instruction is passed and run on the computer.

Compilation happens just once; interpretation occurs each time the program is executed. The figure illustrates how this works.



You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it's a Java development tool or a Web browser that can run Java applets, is an implementation of the Java VM. The Java VM can also be implemented in hardware.

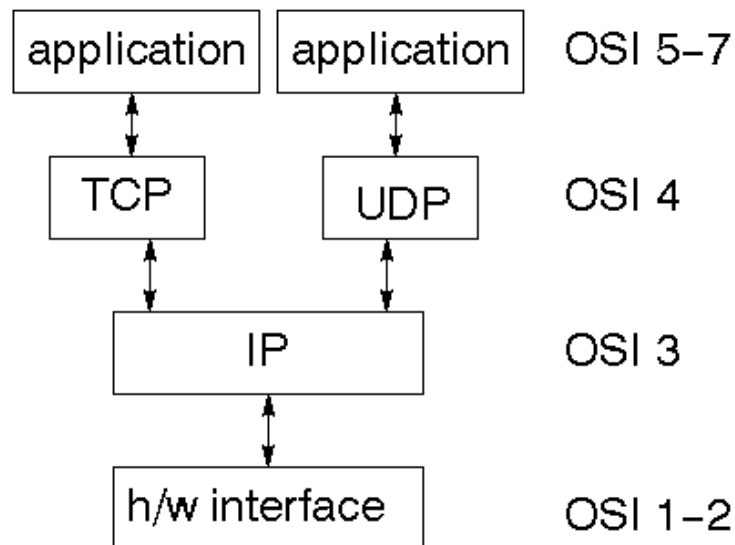
Java byte codes help make “write once, run anywhere” possible. You can compile your Java program into byte codes on my platform that has a Java compiler. The byte codes can then be run any implementation of the Java VM. For example, the same Java program can run Windows NT,

Solaris, and Macintosh.

C. Networking

❖ TCP/IP stack

The TCP/IP stack is shorter than the OSI one:



TCP is a connection-oriented protocol; UDP (User Datagram Protocol) is a connectionless protocol.

❖ IP datagram's

The IP layer provides a connectionless and unreliable delivery system. It considers each datagram independently of the others. Any association between datagram must be supplied by the higher layers. The IP layer supplies a checksum that includes its own header. The header includes the source and destination addresses. The IP layer handles routing through an Internet. It is also responsible for breaking up large datagram into smaller ones for transmission and reassembling them at the other end.

❖ **UDP**

UDP is also connectionless and unreliable. What it adds to IP is a checksum for the contents of the datagram and port numbers. These are used to give a client/server model - see later.

❖ **TCP**

TCP supplies logic to give a reliable connection-oriented protocol above IP. It provides a virtual circuit that two processes can use to communicate.

❖ **Internet addresses**

In order to use a service, you must be able to find it. The Internet uses an address scheme for machines so that they can be located. The address is a 32 bit integer which gives the IP address. This encodes a network ID and more addressing. The network ID falls into various classes according to the size of the network address.

❖ **Network address**

Class A uses 8 bits for the network address with 24 bits left over for other addressing. Class B uses 16 bit network addressing. Class C uses 24 bit network addressing and class D uses all 32.

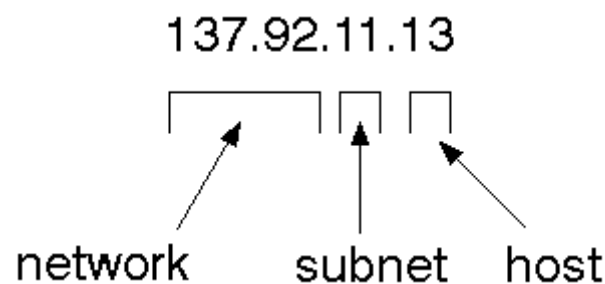
❖ **Subnet address**

Internally, the UNIX network is divided into sub networks. Building 11 is currently on one sub network and uses 10-bit addressing, allowing 1024 different hosts.

❖ Host address

8 bits are finally used for host addresses within our subnet. This places a limit of 256 machines that can be on the subnet.

❖ Total address



The 32 bit address is usually written as 4 integers separated by dots.

❖ Port addresses

A service exists on a host, and is identified by its port. This is a 16 bit number. To send a message to a server, you send it to the port for that service of the host that it is running on. This is not location transparency! Certain of these ports are "well known".

❖ Sockets

A socket is a data structure maintained by the system to handle network connections. A socket is created using the call `socket`. It returns an integer that is like a file descriptor. In fact, under Windows, this handle can be used with Read File and Write File functions.

```
#include <sys/types.h>
#include <sys/socket.h>
int socket(int family, int type, int protocol);
```

Here "family" will be `AF_INET` for IP communications, protocol will be zero, and type will depend on whether TCP or UDP is used. Two

processes wishing to communicate over a network create a socket each. These are similar to two ends of a pipe - but the actual pipe does not yet exist.

❖ JFree Chart

JFreeChart is a free 100% Java chart library that makes it easy for developers to display professional quality charts in their applications. JFreeChart's extensive feature set includes:

- A consistent and well-documented API, supporting a wide range of chart types;

- A flexible design that is easy to extend, and targets both server-side and client-side applications;

- Support for many output types, including Swing components, image files (including PNG and JPEG), and vector graphics file formats (including PDF, EPS and SVG);

JFreeChart is "open source" or, more specifically, free software. It is distributed under the terms of the GNU Lesser General Public Licence (LGPL), which permits use in proprietary applications.

1. Map Visualizations

Charts showing values that relate to geographical areas. Some examples include: (a) population density in each state of the United States, (b) income per capita for each country in Europe, (c) life expectancy in each country of the world. The tasks in this project include:

- Sourcing freely redistributable vector outlines for the countries of the world, states/provinces in particular countries (USA in particular, but also other areas);

- Creating an appropriate dataset interface (plus default implementation), a rendered, and integrating this with the existing XYPlot class in JFreeChart;

- Testing, documenting, testing some more, documenting some more.

2. Time Series Chart Interactivity

Implement a new (to JFreeChart) feature for interactive time series charts --- to display a separate control that shows a small version of ALL the time series data,

with a sliding "view" rectangle that allows you to select the subset of the time series data to display in the main chart.

3. Dashboards

There is currently a lot of interest in dashboard displays. Create a flexible dashboard mechanism that supports a subset of JFreeChart chart types (dials, pies, thermometers, bars, and lines/time series) that can be delivered easily via both Java Web Start and an applet.

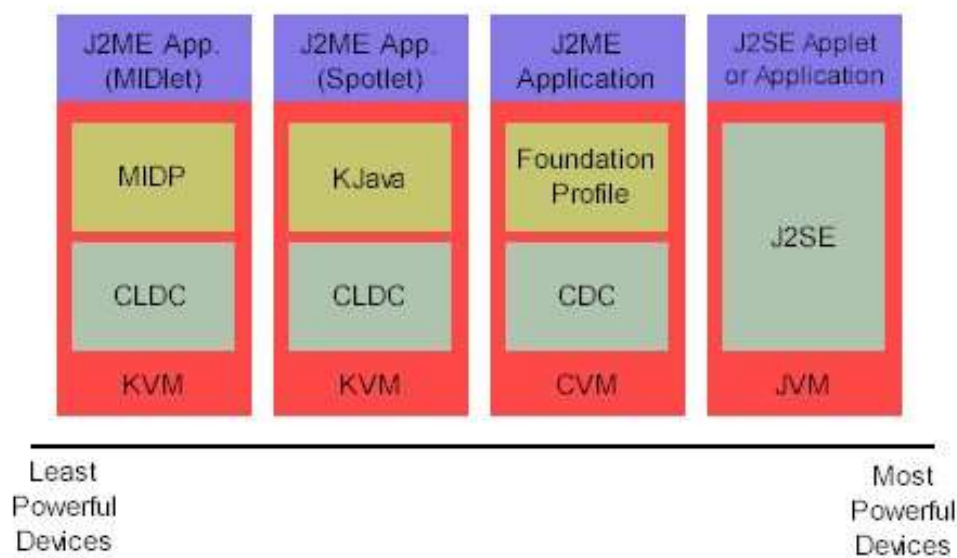
4. Property Editors

The property editor mechanism in JFreeChart only handles a small subset of the properties that can be set for charts. Extend (or reimplement) this mechanism to provide greater end-user control over the appearance of the charts.

❖ J2ME (Java 2 Micro edition):-

Sun Microsystems defines J2ME as "a highly optimized Java run-time environment targeting a wide range of consumer products, including pagers, cellular phones, screen-phones, digital set-top boxes and car navigation systems." Announced in June 1999 at the JavaOne Developer Conference, J2ME brings the cross-platform functionality of the Java language to smaller devices, allowing mobile wireless devices to share applications. With J2ME, Sun has adapted the Java platform for consumer products that incorporate or are based on small computing devices.

1. General J2ME architecture



J2ME uses configurations and profiles to customize the Java Runtime Environment (JRE). As a complete JRE, J2ME is comprised of a configuration, which determines the JVM used, and a profile, which defines the application by adding domain-specific classes. The configuration defines the basic run-time environment as a set of core classes and a specific JVM that run on specific types of devices. We'll discuss configurations in detail in the The profile defines the application; specifically, it adds domain-specific classes to the J2ME configuration to define certain uses for devices. We'll cover profiles in depth in the The following graphic depicts the relationship between the different virtual machines, configurations, and profiles. It also draws a parallel with the J2SE API and its Java virtual machine.

2. Developing J2ME applications

Introduction In this section, we will go over some considerations you need to keep in mind when developing applications for smaller devices. We'll take a look at the way the compiler is invoked when using J2SE to compile J2ME applications. Finally, we'll explore packaging and deployment and the role preverification plays in this process.

3.Design considerations for small devices

Developing applications for small devices requires you to keep certain strategies in mind during the design phase. It is best to strategically design an application for a small device before you begin coding. Correcting the code because you failed to consider all of the "gotchas" before developing the application can be a painful process. Here are some design strategies to consider:

- * Keep it simple. Remove unnecessary features, possibly making those features a separate, secondary application.
- * Smaller is better. This consideration should be a "no brainer" for all developers. Smaller applications use less memory on the device and require shorter installation times. Consider packaging your Java applications as compressed Java Archive (jar) files.
- * Minimize run-time memory use. To minimize the amount of memory used at run time, use scalar types in place of object types. Also, do not depend on the garbage collector. You should manage the memory efficiently yourself by setting object references to null when you are finished with them. Another way to reduce run-time memory is to use lazy instantiation, only allocating objects on an as-needed basis.

4.Configurations overview

The configuration defines the basic run-time environment as a set of core classes and a specific JVM that run on specific types of devices. Currently, two configurations exist for J2ME, though others may be defined in the future:

- * **Connected Limited Device Configuration (CLDC)** is used specifically with the KVM for 16-bit or 32-bit devices with limited amounts of memory. This is the configuration (and the virtual machine) used for developing small J2ME applications. Its size limitations make CLDC more interesting and challenging (from a development point of view) than CDC. CLDC is also the configuration that we will use for developing our drawing tool application. An example of a small wireless device running small applications is a Palm hand-held computer.

* **Connected Device Configuration (CDC)** is used with the C virtual machine (CVM) and is used for 32-bit architectures requiring more than 2 MB of memory. An example of such a device is a Net TV box.

❖ J2ME profiles

What is a J2ME profile?

As we mentioned earlier in this tutorial, a profile defines the type of device supported. The Mobile Information Device Profile (MIDP), for example, defines classes for cellular phones. It adds domain-specific classes to the J2ME configuration to define uses for similar devices. Two profiles have been defined for J2ME and are built upon CLDC: KJava and MIDP..

A skeleton profile upon which you can create your own profile, the Foundation Profile, is available for CDC.

Profile 1: KJava

KJava is Sun's proprietary profile and contains the KJava API. The KJava profile is built on top of the CLDC configuration. The KJava virtual machine, KVM, accepts the same byte codes and class file format as the classic J2SE virtual machine. KJava contains a Sun-specific API that runs on the Palm OS.

Profile 2: MIDP

MIDP is geared toward mobile devices such as cellular phones and pagers. The MIDP, like KJava, is built upon CLDC and provides a standard run-time environment that allows new applications and services to be deployed dynamically on end user devices. MIDP is a common, industry-standard profile for mobile devices that is not dependent on a specific vendor..

MIDP contains the following packages, the first three of which are core CLDC

packages, plus three MIDP-specific packages.

- * java.lang

- * java.io

- * java.util

- * javax.microedition.io

- * javax.microedition.lcdui

- * javax.microedition.midlet

- * javax.microedition.rms

7. Execution

A.WORKING PAGE:

Step:1

Data Owner

- Register , Login
- Browse, encrypt, Upload with mac
- Verify file
- Delete file
- View Search Control & Dec Key Permitted Files

Step:2

Data User

- View My Profile
- Send Search Request
- Search File
- View Cloud Files
- Request secret Key
- Request DAC
- View Access Control Responses
- Download

Step:3

Cloud Server

- View Owners & Authorize,
- View Users & Authorize,
- View Cloud Server Files,
- View Data User Key Access Control Request,
- View Search Access Control Request,
- View Access Policy Details and Its Update,
- View Transactions
- View Attackers
- View File's Rank in Chart
- View Time Delay in Chart
- View Throughput in Chart
- File ID
- File Name
- Owner Name

- Secret Key,

B. RESULT ANALYSIS:

Figure : Then go to

C:\Program Files\Apache Software Foundation\Tomcat 8.5\bin

Directory and run the Tomcat8

» This PC » OS (C:) » Program Files » Apache Software Foundation » Tomcat 8.5 » bin

Name	Date modified	Type	Size
bootstrap	31-03-2022 21:35	Executable Jar File	36 KB
catalina	31-03-2022 21:35	Windows Batch File	17 KB
ciphers	31-03-2022 21:35	Windows Batch File	3 KB
configtest	31-03-2022 21:35	Windows Batch File	2 KB
digest	31-03-2022 21:35	Windows Batch File	3 KB
service	31-03-2022 21:35	Windows Batch File	9 KB
setclasspath	31-03-2022 21:35	Windows Batch File	4 KB
shutdown	31-03-2022 21:35	Windows Batch File	2 KB
startup	31-03-2022 21:35	Windows Batch File	2 KB
Tomcat8	31-03-2022 21:35	Application	135 KB
Tomcat8w	31-03-2022 21:35	Application	123 KB
tomcat-juli	31-03-2022 21:35	Executable Jar File	51 KB
tool-wrapper	31-03-2022 21:35	Windows Batch File	5 KB
version	31-03-2022 21:35	Windows Batch File	2 KB

Figure:1.1 Next Step is to Modularize all the source code in there Respective module/Floder

Name	Date modified	Type	Size
A_Wrong_Login	24-11-2017 11:56	Java Server Pages ...	3 KB
Admin	18-07-2019 14:45	Java Server Pages ...	5 KB
adminauth	18-07-2019 14:59	Java Server Pages ...	2 KB
AdminMain	18-07-2019 16:52	Java Server Pages ...	4 KB
Attack	02-12-2017 12:34	Java Server Pages ...	4 KB
Attack1	02-12-2017 12:45	Java Server Pages ...	6 KB
Attack2	02-12-2017 12:44	Java Server Pages ...	7 KB
authorityauth	31-12-2015 13:41	Java Server Pages ...	2 KB
C_Gautonize	18-07-2019 17:18	Java Server Pages ...	1 KB
C_Process	18-07-2019 17:18	Java Server Pages ...	3 KB
C_Sautonize	18-07-2019 17:18	Java Server Pages ...	1 KB
C_Uautonize	18-07-2019 17:18	Java Server Pages ...	1 KB
C_ViewAttackers	18-07-2019 13:28	Java Server Pages ...	5 KB
C_ViewAudit	18-07-2019 13:24	Java Server Pages ...	5 KB
C_ViewDAC	18-07-2019 17:09	Java Server Pages ...	5 KB
C_ViewFiles	18-07-2019 13:24	Java Server Pages ...	5 KB
C_ViewOwner	18-07-2019 13:24	Java Server Pages ...	7 KB
C_ViewRank	02-12-2017 14:36	Java Server Pages ...	3 KB
C_ViewReq	18-07-2019 13:26	Java Server Pages ...	6 KB
C_ViewSearchReq	18-07-2019 13:14	Java Server Pages ...	5 KB
C_ViewTD	02-12-2017 14:36	Java Server Pages ...	3 KB
C_ViewTH	02-12-2017 14:37	Java Server Pages ...	3 KB
C_ViewTransac	18-07-2019 13:26	Java Server Pages ...	5 KB
C_ViewTransactions	18-07-2019 13:27	Java Server Pages ...	5 KB
C_ViewUsers	18-07-2019 13:25	Java Server Pages ...	7 KB
cauditauth	31-12-2015 13:43	Java Server Pages ...	1 KB
cdatasauth	02-12-2017 12:18	Java Server Pages ...	1 KB
chart_bg	28-02-2011 10:18	JPG File	6 KB
cloudauth	31-12-2015 13:00	Java Server Pages ...	2 KB

Figure:1.2 Then go to browser go to localhost:8090

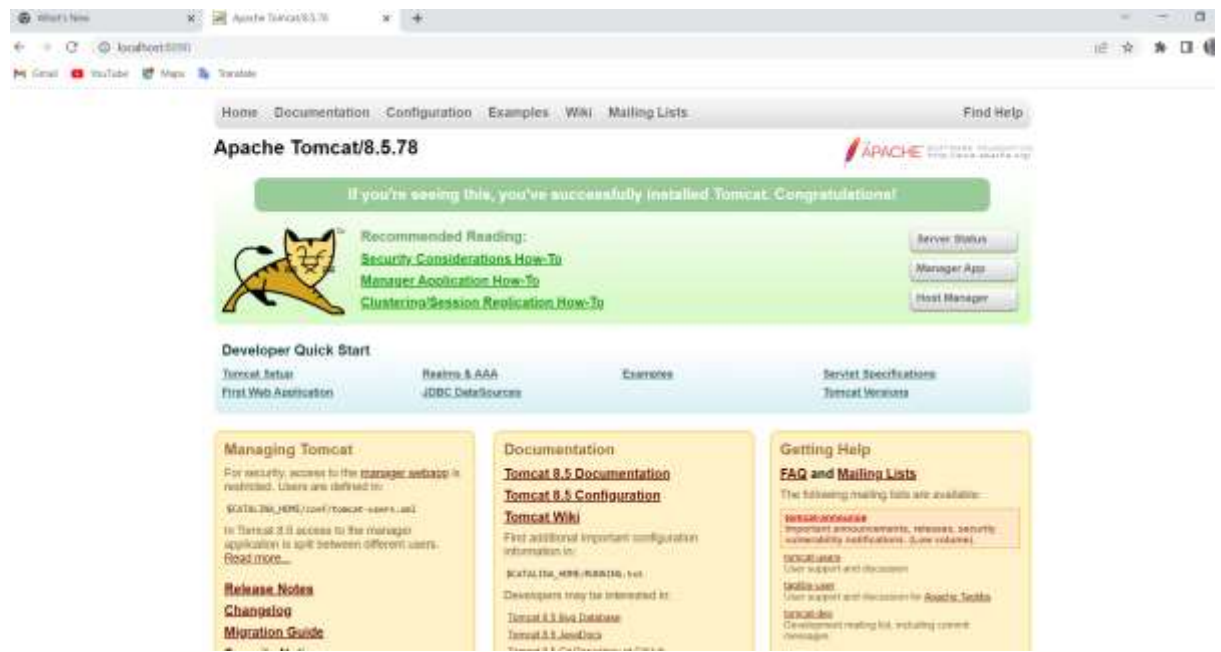


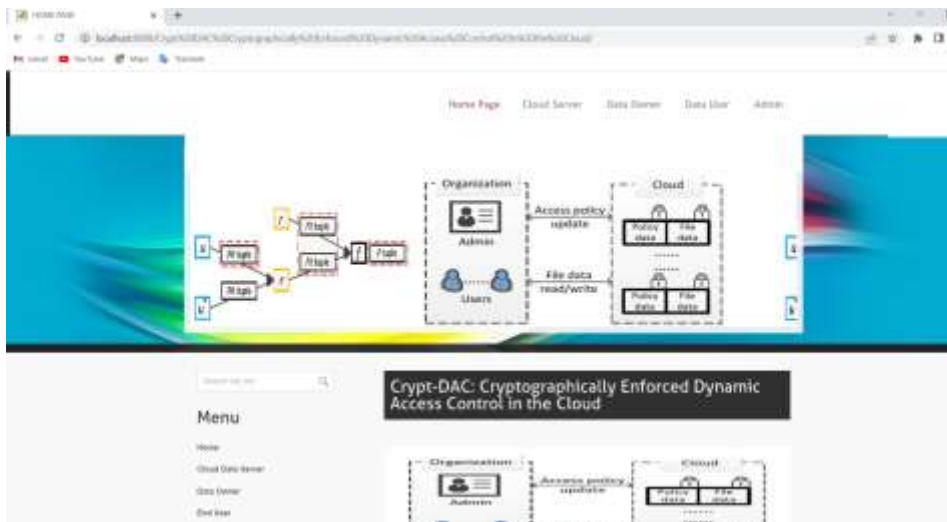
Figure:1.3 then click on Manager App, which

will show us the application running



Tomcat Web Application Manager						
Manager						
URL Descriptions	HTTP Manager Help	Manager Help	Server Status			
Applications						
Path	Version	Display Name	Running	Sessions	Comments	
/	None specified	Welcome to Tomcat	true	0	Start Stop Reload Undeploy [Copy session with size 0] remove	
/cloud/	None specified	A fully distributed scheme with Guaranteed Quality of Service in Cloud Computing	true	0	Start Stop Reload Undeploy [Copy session with size 0] remove	
/docs/	None specified	Tomcat Documentation	true	0	Start Stop Reload Undeploy [Copy session with size 0] remove	
/examples/	None specified	Servlet and JSP Examples	true	0	Start Stop Reload Undeploy [Copy session with size 0] remove	
/host-manager/	None specified	Tomcat Host Manager Application	true	0	Start Stop Reload Undeploy [Copy session with size 0] remove	
/manager/	None specified	Tomcat Manager Application	true	1	Start Stop Reload Undeploy [Copy session with size 0] remove	

Figure :1.4 then click on our web application



Let's explore the web page now, the Data Owner, End user section will act as the login name and Password. First-of-all we should Register the page then Data owner granted to permission to login page.

Figure: 2.1 Data owner Login page

Figure:2.2 Data owner have to Register to the page and have to wait till Admin verifies id.

Figure:2.3 After when the Data Owner is verified by the Admin, User can login through username and password.

Working:

To Work with our Website/web Application first we have to create our account(user/Register here)

And also,we can work with the asmin usin(admin/admin) username and password

Figure:2.4The user has to login with valid user name and password

Search our site

Data Owner Login

Menu

- Home
- Cloud Data Server
- Data Owner
- End User

Name (required)
panday

Password (required)
.....

[Register](#) [Submit](#)

Figure: 3.1 This is user owner home page here user can,view my profile,search secure file,upload file and delete file



View My File

Owner Image	Owner Name	E-Mail	Mobile	Address	DOB	Location	Status
	dataowners	dataowners@gmail.com	9988776655	hyd	09-09-1998	hyd	Authorized

[Go Back](#)

Search our site:

Upload File


Main Menu

[Data Owner Main](#)
[Log Out](#)

Data Owner Menu

[Upload](#)
[View My Files](#)
[View My Profile](#)
[Verify](#)
[Delete File](#)

Select File :-

Choose File


File Name :-

panday

Trapdoor :-

Encrypt

```

***** 00IF 00 d d ***** 00Ducky 00 0 *****
!http://ns.adobe.com/xap/1.0/ <?xpacket begin="
id="W5M0MlpCehiHzreSzNTczkc9d"?>
<x:xmeta xmlns:x="adobe:meta/" x:xmptk="Adobe
XMP Core 4.2.2-c063 53.352624, 2008/07/30-18:12:18
">
  <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-
rdf-syntax-ns#">
    <rdf:Description rdf:about=""
      xmlns:dc="http://purl.org/dc/elements/1.1/"
      xmlns:xmpRights="http://ns.adobe.com/xap/1.0/rights/"
      xmlns:photoshop="http://ns.adobe.com/photoshop/1.0/

```

Verify File

File Name :-	<input type="text" value="panday"/>
	<input type="button" value="Verify"/>

Delete File

File Name :-	<input type="text" value="panday"/>
	<input type="button" value="Delete"/>

Figure: 4.1 This was show the cloud permission request from user and data owner

View Data Access Control			
User Name	Request Date and Time	Read Data	Write Data
Sumit	18/07/2019 15:55:51	Permitted	Permitted
nikhilaanj	19/07/2019 11:46:08	Permitted	Permitted
Vishnu	19/07/2019 11:52:33	Permitted	Permitted
dataowner	06/05/2022 10:15:00	Permitted	Permitted
hitesh	13/05/2022 12:46:13	Permitted	Permitted

[Go Back](#)

Figure: 4.2

This show the view all the user request to cloud permit

View User Search Request & Permit		
User Name	Date and Time	Status
Sumit	18/07/2019 15:02:01	Yes
nikhilaanj	19/07/2019 11:46:08	Yes
Vishnu	19/07/2019 11:52:30	Yes
dataowner	06/05/2022 10:15:00	Yes
hitesh	13/05/2022 12:46:13	Yes

[Go Back](#)

• **SYSTEM TESTING**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPES OF TESTS

➤ **Unit testing**

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

➤ **Integration testing**

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

the combination of components.

➤ **Functional test**

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

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.
Invalid Input : identified classes of invalid input must be rejected.
Functions : identified functions must be exercised.
Output : identified classes of application outputs must be exercised.
Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

➤ **System Test**

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

➤ **White Box Testing**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

➤ **Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it.

- ## **INVESTIGATION**

The first and foremost strategy for development of a project starts from the thought of designing a mail enabled platform for a small firm in which it is easy and convenient of sending and receiving messages, there is a search engine ,address book and also including some entertaining games. When it is approved by the organization and our project guide the first activity, ie. preliminary investigation begins. The activity has three parts:

- **Request Clarification**
- **Feasibility Study**
- **Request Approval**

REQUEST CLARIFICATION

After the approval of the request to the organization and project guide, with an investigation being considered, the project request must be examined to determine precisely what the system requires.

Here our project is basically meant for users within the company whose systems can be interconnected by the Local Area Network(LAN). In today's busy schedule man need everything should be provided in a readymade manner. So taking into consideration of the vastly use of the net in day to day life, the corresponding development of the portal came into existence.

FEASIBILITY ANALYSIS

An important outcome of preliminary investigation is the determination that the system request is feasible. This is possible only if it is feasible within limited resource and time. The different feasibilities that have to be analyzed are

- **Operational Feasibility**
- **Economic Feasibility**
- **Technical Feasibility**

- **Operational Feasibility**

Operational Feasibility deals with the study of prospects of the system to be developed. This system operationally eliminates all the tensions of the Admin and helps him in effectively tracking the project progress. This kind of automation will surely reduce the time and energy, which previously consumed in manual work. Based on the study, the system is proved to be operationally feasible.

- **Economic Feasibility**

Economic Feasibility or Cost-benefit is an assessment of the economic justification for a computer based project. As hardware was installed from the beginning & for lots of purposes thus the cost on project of hardware is low. Since the system is a network based, any number of employees connected to the LAN within that organization can use this tool from at anytime. The Virtual Private Network is to be developed using the existing resources of the organization.

- **Technical Feasibility**

According to Roger S. Pressman, Technical Feasibility is the assessment of the technical resources of the organization. The organization needs IBM compatible machines with a graphical web browser connected to the Internet and Intranet. The system is developed for platform Independent environment. Java Server Pages, JavaScript, HTML, SQL server and WebLogic Server are used to develop the system. The technical feasibility has been carried out. The system is technically feasible for development and can be developed with the existing facility.

REQUEST APPROVAL

Not all request projects are desirable or feasible. Some organization receives so many project requests from client users that only few of them are pursued. However, those projects that are both feasible and desirable should be put into schedule. After a project request is approved, its cost, priority, completion time and personnel requirement is estimated and used to determine where to add it to any project list. Truly speaking, the approval of those above factors, development works can be launched.

• CONCLUSION

We presented Crypt-DAC, a system that provides practical cryptographic enforcement of dynamic access control in the potentially untrusted cloud provider. Crypt-DAC meets its goals using three techniques. In particular, we propose to delegate the cloud to update the policy data in a privacy-preserving manner using a delegation-aware encryption strategy. We propose to avoid the expensive re-encryptions of file data at the administrator side using an adjustable onion encryption strategy. In addition, we propose a delayed de-onion encryption strategy to avoid the file reading overhead. The theoretical analysis and the performance evaluation show that Crypt-DAC achieves orders of magnitude higher efficiency in access revocations while ensuring the same security properties under the honestbut curious threat model compared with previous schemes.

