2. JAVA TECHNOLOGY AND LITERATURE SURVEY

2.1 Java Technology

This complete section of the thesis gives the whole details about the Java language security which is a highly great software development technology. All has been described in the coming discussions.

2.1.1 Java Language Features

Platform Autonomy (Portable): It is the methodology named as WORAW which is well understood by us as an independent to the computer system configuration. It is the best property of the Java language. Other languages are not achieving this feature; this is the one which gained a lot to be very nearer to this property. All the programs which are prepared on the Java platform can be executed on any different platform with the help of JVM (Java Virtual Machine).

Very Easy: There are diverse marks that make it as a straightforward programming language. Programming is so comfortable as well as to be tested easily because Java not explicitly uses the pointers.

Object Oriented: Around an object to be, that language is oriented, some language has to go after the successive 4 fundamental concepts.

- **Inheritance**: Concept tells the reusability of the system. Be reused the procedure to create of novel classes as well as use manners present in the available classes through expanding it only the accessible code and the adding of the extra marks when necessary.

- **Encapsulation**: In it, the data and their functionality are wrapped up under one cover and then furnishing of the abstraction.

- **Polymorphism**: It is the word which is taken from the Greek language. Here “poly” means many, and “morph” means forms. From one name we can do lot of functioning according to the situations. Or we can say “it is the technique to furnish of the diverse functionalities through the methods which is based on the same name but different on the signatures of the methods”.
- **Dynamic binding:** Occasionally we are not aware about the objects and their particular working, when we go for the programming. This is the technique to furnish the maximal functionality to the programs at the time of their execution.

There are so many languages like Objective C, C++ which accomplish all the mentioned properties. But so far these are not to be called as object oriented one completely. The reason behind is that they are procedural plus object oriented. However, the Java that is completely an object-oriented language. It is because the objects are the ones which are the external level of the data structure, as we talk about the Java language.

**Robust:** There is the powerful storage allocation and the rubbish collection mechanism which is automatic in Java language. There is the strong exception handling system and strong type checking system in it as we contrast with the other available languages. The compiler ensures the written programs to be error free.

**Distributed:** In the current scenario the mostly used protocols are the main internet protocols (like FTP, HTTP, etc.). Both of the protocols are created in the Java language. The networks programmers call the functions on these protocols. And they can utilize the files which are stored on the distant computers which are the part of the network. There is no need to write the program code on the local computer.

**Dynamic:** In the carrying out the Java programmed the consumer can get away the required files dynamically of a local drive or of a computer thousands of miles of the consumer only through connecting with the Internet.

**Safe:** Java does not use storage pointer explicitly. The entire programs in java ran under a zone well known as the sand box. Security manager intends the attainability option of a class such as harvests and letter of a file for the local disk. The byte code verifier tests the classes after loading.

**Performance:** Java utilizes indigenous code practice. The threads are used to share code and data sections. At the very start of the interpretation of byte code results, the performance was slow. But, the most recent versions of JVM utilizes according to the environment. Now the compilation has increased the performance of the Java programs.
Multithreaded: With the help of Java programming language, we can also do the multithreaded programming. Multithreading is a technique in which a single program has dissimilar threads. These threads are separate one which is carried out concurrently. The instructions of the numerous threads are executed regarding the program. There may be many threads under one process (a program in execution) of the computer. Multithreading analogy is shown as many processes execute on a single computing machine. The programming of multithreading has a lot of curiosity of the developers.

Interpreted: We know very well that Java language has both the facilities of translator. These facilities are to interpret and to compile. With the help of interpreter, we can execute the programs from their source code directly. The on-the-fly translator (the interpreter) reads the source code instructions one by one. So, Java is a good programming language to interpret and due to it is very easy to find out the errors from the new programs.

Architecture Neutral: The concept architecturally neutrally seems to be eccentric, but Java language is an architectural neutral one. Networking has gained reputation among people. It has given the idea of distribution on the networks to the programmers. Today is the world of networks in which so many computers are interconnected from distant. So we can move around the applications easily to separate computer machines. The applications can be distributed on diversity of computer architectures and diverse number of operating systems. The Java language programs are converted to the byte code after the compilation and that code is then interpreted on any type of machine. The machine must have the Java Virtual Machine.

2.1.2 Java Security Basics

Here are two types of security in the Java language that are as follows:

Security at Language level

Entire security is put through some systems that are built up at diverse occasions. The bases of the java security can be seen in several fundamental language marks:

- Has and easily simplifies, to use. Java is a large amount straightforward to compare to new languages such as C++. Consequently the load is on the developer is more less and so the likelihood to make of fine mistakes is diminished;
Finalization of classes and routines. This mark does not let sub classing to if used to incorporate definitions, and canceling lets when for method definitions counted doesn't have to, and the unwanted change of that of functionality firm stops object oriented;

Severity. There are packaging classes that even for the straightforward data types, and it defined become, can give no constructions exterior of classes. Consequently all securities told advantages of the object oriented pattern example can be utilized;

Automatic storage management: The pointers are not directly used here and there is no arithmetic for the memory addresses. It does not allow wrong storage accessing and has the less likelihood of the leakage of memory. There is no trespasser data admittance and execution time slumps;

Physically powerfully typed minimizes. Polymorphism is a extremely influential object oriented mark, but it embraces possible risks to disguise of aggressive objects. Both the compiler and the execution time testing does not allow such opportunities since no task can be made, if against types objects, there is not compatibility;

More severe exception treating mechanism is. At the moment of method use, there may be heavy mistakes; at that time the developer is compelled by the Java compiler to treat all types of exceptions. Consequently the manners of the Java code are unsurprising the program code turns into "fool-proof".

Defined clear performance to uninitialized of variable. All on pile, that involuntarily is been based memory, is initialized. But all on pile, that is been based memory, is not. So all class and case variable never are set to indeterminate values, and all the local variables must be allotted certainly before exercise or the source compiler is forced to give you a mistake.

Java is a revolutionary new application platform of Sun Microsystems. Such as other operating systems, platform developer with I/O furnishes the java, must networking, window and graphics productivity and other arrangements and develops to run refined applications. Also an important productivity that is not found in traditional operating systems furnishes the java platform. This capacity, is called that once/course somewhere feasible files writes, permits java program, that be written on a kind of hardware, or operating system to run unchanged on almost other kind of computer.

**Virtual Security at Machine Level in Java**
The JVM (Java Virtual Machine) is answerable for the running of Java program codes. It is also answerable for the implementation of much vital division of the Java Security properties:

- **Byte-code confirmation:** The platform independent code is generated by the Java compiler than can be executed on the JVM. The byte code verifier and the compiler guarantee the legal byte code for Java is executed. The language security is the responsibility of the both the compiler as well as the byte code verifier.

- **Class loading dynamically:** It is the duty of the class loader to seek and freight the byte code for the specified definitions of the classes. The class Loader finds and loads the byte codes for the class definitions. After load, they are attested ahead of the formation of real classes. The class Loader assures the JVM that it is not trapped in the fake illustrations of the main libraries of the classes. It can break the Java Security Model. At the end, the class loader offers to split the namespaces for the classes which are packed from diverse locations. It thwarts the classes which are not the trusted ones that can interface with the other executing programs.

- **Execution Time Security Checks:** The dynamic binding is granted with the help of JVM. There is added execution time type checking is always of assignment statements and array boundaries. Due to this, JVM assures the correct assignments to the operations and arrays are not accessed external to their bounds.

- **Control decisive system resources over the admittance to:** Every working JVM that is installed, has mainly single Security Manager. Each statement is cross checked by the Security Manager in the advance. It stops the activities done by the unfaithful code.

- **Permissions and Access Controller:** The security managers cannot be installed by the applications. But it is feasible to offer personal permissions for the particular activities to the applications. It is done with the help of the Policy Tool program. The permissions are easily checked by with the help of AccessController class to do an activity. The JVM’s security implementations are described with the help of the “Sandbox”. In it all the programs of Java language runs securely. This is completely risk free for the system as well as to the users. The sandbox is coming with Java’s very first version.

Those compelled I am to hit slightly people in charge of across these rules the compiler, the byte of code of verifier and that really the method of Java (JVM). At the very beginning the compiler is the defensive. At the time of compilation the rules are cross checked. The compiler
cannot compel to check the array boundaries and the cases of unlawful casts. All the checks are completed at the execution time only. The casting becomes a headache when the objects are not the related ones.

```
Object ab = mVector.elementAt(0);
Bus lmn = (Bus) abus;
```

The compiler has no technique to understand that the object arrived from vector is a bus, or it is behaving like a bus. At the time of loading of the classes in Java, the verifier for the byte code offers a technique to cross check the rules that are given. More of those first admissions in the list, it is reassured by the byte code verifier that is as follows:

- The correct format of class file,
- There is only a single super class for each derived class (Multiple Inheritance is not supported by the Java).
- Operands are not with underflow or overflow.

The byte code verifier and the compiler of Java have the mixed responsibilities which are as the similar checks are executed. It is the necessity to do double check for the untrusted code of third persons. We use the verify option as a command line argument by the byte code verifier, when we execute the program. At the time of future releases of Java system development kit, the byte code verifier will run by default for better security. But recently we have been recommended to use the verify option for third person’s programs. The JVM is answerable to check the array boundaries and the validations of object casts. The java.lang.ArrayIndexOutOfBoundsException is used for the checking the boundaries at the execution time.

**Sandbox**

In the Java 2 SDK, the sandbox model is initiated. It enhances the security in the controlled access and the implementation of a new attitude. In it the applications are executed with diverse permissions. There is no fixed notion for the trusted code. This sandbox assures:

- Security model is extended dynamically;
- Resources are protected fine-grained access control;
- Security manager enforces the security policy very easily;
The access control structure is easily expandable;

Extended security checks for all the programs in Java with applets.

---

The independent programs in Java always run by default, at that time no sandbox is activated. It is activated with the help of command line option for executing the applications as `Djava.security.manager`. The concept says that there is a need for the basic elements for setting up the sandbox.

- There is a need for the predefined set of susceptible actions.
- These are bound with the actions those are specified in the code segment.
- There is a control center which is answerable to allow or disallow the code to perform specific operations.

Permission: There is an abstract class named as `java.security.Permission` in Java. Each class is belonged to a permission set. Any class in Java can do several activities. Any class that belongs to the specific class named as `java.security.AllPermission` can do everything. Every permission has three basic properties.

- **Type**: It is the name of the specific class in Java that implements a permission.
- **Name**: On the basis of the permission type, a name belongs to permission to a file. It is the aimed directory or a file. There is no name entry for much permission.
Actions: It is the optional one that describes the entry lists. There may be file permissions like read, write, or delete one.

In the directory “/Users/public/shared/”, the permission for reading a file is described as follows:

```
Permission java.security.FilePermission
="/User/public/shared/", “read”
```

Here the type of the permission is java.security.FilePermission. It’s name is /User/public/shared. Its action is read.

Sources of code: It encapsulates the information regarding the class that is loaded and who has marked the specific class. All these are elective. The description of location is told with the help of URL (Uniform Resource Locater), code base. All the source code is defined in Java with the java.security.CodeSource class. The information regarding the marking is wrapped in a table named as java.security.cert.Certificate instance. The public key is the identity’s advantage to bind the object certificates.

Protection Domain describes the relationship within the code sources and the permissions. It means what the source code will do. It binds together the code and the particular permissions for them to do. The java.security.ProtectionDomain class has the instance of a protection domain. Every class corresponds to a protection domain.

Policy files give the details regarding protection domains. These correspond to code source to permissions. Whichever is number of these files can be used by the JVM. By default this number is two.

The global policy file is located in the directory named as “$JREHOME/lib/security/java.policy”. It gives the directions for the root catalog the installations of Java.

The user specific policy file is located in the directory named as “${user.home}/.java.policy” or simply in “${user.home}”. It describe the home catalog of the recent user.
An extort of the global policy file:

```java
grant rootBase file:${java.home}/lib/ext/*
{
Permissions.java.Security.AllPermissions;
};
```

Here `${java.home}` shows the root catalogue of Java installation.

The outcome is that the source of codes $java.home/lib/ext is connected with the particular AllPermission, that means, that code can do of this indicator, what wants also always it.

The `java.security.Policy` is the abstract class. It is implemented through programming. It wraps up the policy files that are in utilization. It is system security policy. The JVM can have the only instance of the policy class on a particular time. However it can be changed if we have the permission. We can direct the particular policy file that is used by the system in the “$JREHOME/lib/security/”. The policy file can also be changed with the help of command line. It is the optional one.

```java
java.security.policyfile = ownPolicy.policy
```

Now we can use the policies with the help of three choices which are as global one, command line based one, or user specific one. In the command line we will use the two symbols of equal (==) to activate the one and only one.

In to summarize, we have the choice of a specific class to be permitted with the use of source code and the permissions.

The access controller is used to determine a class that has the permissions to execute the susceptible operations. The fundamental operating systems and the class of policy file should communicate with each other. The access controller manages and controls all the security policies of all the applications. Before the Java2 it was done with the help of security manager. The method was available with us but now it is very easy on Java 2 platform. The scenario is common now, but the access controller is always different from the security manager in the actions. The access controller is enhanced because it is better in fine grained way for the security
policies. It is very difficult to implement all the policies in the security manager of the operating system.

Access Controller resides in the class named as “java.security.AccessController”. It has no instance. However it has static methods in it which are used to query the security policy state in cause. We can use the methods for checking the specific permissions which are allowed.

“Public Static void checkPermission (Permission p)”; 

The access controller has an exception named as “AccessControlException”. It is thrown if the particular operation is not permitted else the method is returned normally. The operation will be continued to do. The stack of the protection domain set take this decision. The stack contains the references of the live classes in the specified thread. It is done with the help of the per-thread basis scheme. It means that every thread has a stack in the application. The Figure 2.1 describes the organization of the stack. It is done when the checkPermission (p) method is invoked by the Access Controller class object. All the recent permissions are shown through the crossing of the permissions which are related with the protection domain. This relation is shown in the figure with the help of double arrow.

As we know that the classes which are loaded from the core Application Programming Interface are the identifiers of the system domain. The core API can do every type of operations.

All the basic vital permissions are related with the loaded classes, anywhere. In general these are the other domains (other than the system domain). When the method checkPermission (p) is called the stack of the permissions is traversed from the top to bottom. This checking is well shown in the Figure 2.1 with the arrow numbering. The cross section of the protection
domain describes the recent policy of the security. In the Figure 2.2, we have shown the association within the vital classes of the Java security package. It tells the working of the access controller.

The objects of the class java.security.AccessController contains all the references of the recent policies of the security. It is wrapped up in the object of the class java.security.Policy. At the random time, there is always a single policy for the application. The polices can be accessed with the invocation to a static method of the policy class named as getPolicy( ). The policy class describes all the permissions which are accessible for the code from diverse sources. The access controller searched out all the results through the system policy class. The entire needed protection domain is contained with the information regarding all the permissions that are allotted to the particular source code. It is done when the operations are the sensitive one only.

To exemplify the process that is represented in the Figure 2.3, we suppose the following time events: A computer programmer desires to discover if her program will become permitted to utilize, to connect a random host on the network. The following section of code will do the trick:

```java
SocketPermission term = new SocketPermission("*:1-", "connect" you);
AccessController.checkPermission(term);
```
Figure 2.3: AdmittanceController class hierarchy

The ([hostname]:[port the area to], [action]) is shown by the cords as they went nearby the SocketPermission’s manufacturer. In this instance of the makeup joker "*" the means break up indicates all of the probable hostnames, the makeup ":" the hostname from the region of seaport and "1" -every one of the seaports of 1 and on the way to the peak. The worth "colleague", points to ours inquire should be permitted to position several connections to house. 

Next call was made to the checkPermission, that (ProtectionDomain, persisting it) the scheme riveted turn into the class of create for see asked if the given consent is hold in the at hand begin of the system. To resolve if it is, the command of security riveted (persisting it) the scheme is described. At the end of the object of PermissionCollection, it can do the different permission of containers, is described,

The class loading mechanism of the byte code is the accountable change in Java that is the freighter of the specified class. All this is done with the help of the JVM and simultaneously with the byte code. For the security purposes, the class freighter plays an important role due to the following reasons:

1. Collaborated namespace must be defined with the JVM.
2. The specific class is permitted for the definition to call the security manager.
3. The permission sets are coordinated up to include the objects.

A sole namespace has been given to each class of the Java language. It is a packet that belongs to the namespace and the name of the class is allotted. There is the class of Mathematics named as java.lang.math namespace. A namespace is defined and determined under the idea to defend the product of security on the platform of Java. The namespaces are not differentiated among the others by the JVM. They create the confusion across the provider of its original execution of the classes of the API of Java. It creates the turmoil. The freighter understands the right version of the given namespace.
The answerability of the class loader is well shown in the Figure 2.4. First of all the class loader consults the security manager to check out the required permissions for the new operation to carry, when the loading of the new classes is prompted. The exception is thrown when there is a failure to represent the valid access permission. Under label 1, we have shown the intersection. For defining and accessing of classes, we require two of the permissions. These are defineClassInPackage and accessClassInPackage.

The sufficient protection domain is created by the class loader, when any class is loaded. It is the mapping of particular source code for one or many permissions. It is well shown in the Figure 2.4. The policy file resides in it all the protection domains for the needed information. As the mapping is shown, it lets the access controller to understand which class has what type of permissions. In this manner, we can describe our new security policies with the help of custom class loader. It is believed that it has low headache than implementing a java.security.Policy class.

There is a hierarchy of class loaders. There is a main class called as system class loader. All the Java Application Programming Interface classes are loaded by this class. There is more than one successor such as URL class loader which is answerable for loading other classes from the classpath. Class loaders are always consulted to their ancestor classes for prompting the new classes. It is a recursive, and nonstop action. It makes the ancient class loaders for the questioning. When no class is found, then the chain of calls is backed down.

The Java Security Model is shown in the figure 2.5. There are three flavors of source files which are as core API, local, and remote. The core API files are translated into the byte
code by the language translator of Sun Microsystems. These are shipped with the Java Runtime Environment (JRE). Then the complete code is considered well trusted. So there is no need to byte code verification. The local code is compiled by the local compilers. The class files are usually remotely loaded resources. It means that the source files have been translated by the remote computer compilers which are non-trustworthy ones. It is ensured that the code follows the syntax and semantic of the Java programming language. All these files are cross checked by the byte code verifier. And then these are loaded in the Java Virtual Machine (JVM). The byte code verification is done for the local classes, as matter of choice. But it is greatly recommended to do the verification. The direction of the arrow is from the remote class files to the local class files in the Figure 2.5. It shows us the scenarios of downloading of the remote class files and stores them on the local hard drive. It makes the byte code verification as a necessity for the better security. We must have the files to be trusted ones. The core API class files are loaded in the JVM. It is done with the help of system class loader.

The class loaders in the down hierarchy of the figure load the entire additional class files. The diverse number of loaders co-operate in the design of sole namespaces for the classes. The classes are loaded in the virtual machine.

The Java Security Model:

The arrow direction in between the Java Virtual Machine and the Operating System demonstrates that the Virtual Machine (JVM) take action on resources. The primary host controls these resources. This admittance is mean to the rules, that are defined for the consumer, who

![Figure 2.5: The Java security model](image)
carries out the code, that someone registered in the OS with supervisor privileges, make will can restricted, what delights also always it.

2.1.3 Cryptography in Java

Cryptographic facilities in Java language are completely utilized with the help of Java Cryptographic Architecture (JCA). The cryptographic module libraries are investigated in this work as an introductory part. The history tells that the cryptography is included in the version 1.1 of Java. It introduces the JCA.

Additional cryptographic attributes are included in the Java Cryptography Extension (JCE). The Java Cryptographic Extension Application Programming Interface (JCE API) was incorporated in the Java version 1.4 of the Java language platform. In enhancement, the Java Secure Sockets Extension (JSSE) and the Java Authentication and Authorization Service (JAAS) module libraries recommend the services like secure communication using SSL (Secure Socket Layer), authentication, and authorization. These services totally relied on the cryptography.

The Java CertPath Application Programming Interface, and XML Digital Signature Application Programming Interface are the more library modules that are to be added for the boosting of the concept of security in the Java Technology.

JAVA CRYPTOGRAPHIC ARCHITECTURE

A bunch of Application Programming Interfaces (APIs) are included in the Java Technology. These offer so many classes which are useable for writing various secure applications on the Java platform. In the starting some these packages were optional, but in today’s scenario all these are combined together in the Java 2 System Development Kit (Java 2 SDK). All the APIs have very rich set of features. They provide a large range of tasks. The JCA is a skeleton which provides cryptographic capabilities to Java programs. It maintains for message digests, key pairs management, digital signatures, and digital certificates.

The Model-View-Controller (MVC) model is followed in the design of this skeleton. The security providers deliver all the implementations. A set of algorithms are provided in the JCA by the Sun standard security provider. But we can also use the third party security provider library modules.
The Java Cryptographic Architecture provides so many frameworks which are the
generic one, pluggable one, and extendable one for achieving cryptography in Java. At the
design time basically two principles were considered which are as follows.

1. Autonomy in Algorithms, and
2. Autonomy in their Implementations.

The autonomy in algorithms is reached through the definitions of engine classes, while
the engine classes encase themselves with the generic of particular Cryptographic services.
There are basically three options which are available to the programmers. These are as follows:

- Independent work (completely self programming),
- Utilization of the cryptographic library modules which are packed up with the Java Development Kit, and
- Introduce the Java environment from third party software.

The autonomy in the implementation of algorithms is achieved through the architecture
of the system.

JAVA CRYPTOGRAPHIC EXTENSION

JCA extension named as Java cryptography extension (JCE) furnishes the frames and
executions for enciphering, key generation and key agreement, and message authentication code
(MAC) algorithms. It also sustains the sealed objects and secure streams. The Java 2 SDK
comprises the standard JCE. It is named as “SunJCE”. It comes with the System Development
Kit and JCA gives the flexibility that we can easily plug it into the Java Cryptographic Extension. The new algorithms are also possible to be included in the JCA.

This agreement ensures that the set of keys to be used for secure communication. The
data integrity is ensured by the Message Authentication Code (MAC) which is a keyed hash
function. A variety of methodologies are used from the JCE.

Java Secure Sockets Extension (JSSE)

It is the skeleton which is used for the customer based authorization and authentication. It
is a standard type of Pluggable Authentication Module (PAM) structure. It is used for the two
basic reasons which are as follows:
1. User’s reliable and secure authentication: It is basically a reliable and secure decision for which the recent Java code is running. Here we don’t consider, the Java code, it may be application or applet or bean or servelet. The fundamental authentication technology is not considered. JAAS authentication is done in a stoppable way.

2. User’s access control rights: In it the users are authorized to assure that they have the particular permissions to do the actions.

   JAAS strategy expands the Java 2 strategy with the help of appropriate subject oriented information. JAAS regularly identify and be aware of all the consents of Java 2. This protocol can be utilized in combination with the application layer protocols. The HTTP (Hypertext Transfer Protocol) and the File Transfer Protocol (FTP) are the application layer protocols. Another protocol is Transport Layer Security (TLS), it offers the following facilities.
   
   o Common Authentication: The client and the server can set up the faith throughout the utilization of digital certificates.
   
   o Encrypting Data: The data that will travel through the networks to the communicating parties is converted into the incomprehensible form with the help of encrypting algorithms. It assures the isolation and secrecy.
   
   o Integrity of data: It converts the data that is sent over communication media are identified. This is the service which safeguards not in favor of replays of the previous messages.

   JSSE is a skeleton for the implementation of Secure Socket Layer in Java.

**Java Authentication and Authorization Service (JAAS)**

   JSSE (Java Secure Socket Extension) permits the secure communication over the networks. It offers a strong skeleton. The protocols SSL as well as the OSI models transportation film are implemented in the Java version.

   Authentication and authorization are the main subjects of JAAS. In it, the consumers and the services run all the applications with authenticity. The Pluggable Authentication Module Framework (PAMF) splits the application programming code from the primary authentication. The authentication is supported on a Java implementation of the PAMF. The authentication technology is replaced down to plugging in the needed explanation.

**Java Generic Security Services (JGSS)**
The JGSS Application Programming Interface encloses all the fastenings for GSS-API. It is a consistent application programming interface for common authentication of the server and the client for the exchanging of information securely. We are not in the concern of the basic technology.

The JGSS-API provides a lot of common characteristics as provided by the JSSE. But, there are so many variations in the basic mechanisms of the security. Here we utilize the Kerberos version 5 in spite of Secure Socket Layer and Transport Layer Security protocols which are used in its counterpart (JSSE). The complete communication API of the JGSS-API is token-oriented in spite of socket-oriented. The credential delegation and selective encryption are also utilized by the JGSS-API. These are not suitable for its counterpart JSSE.

**Java CertPath API**

This application programming interface includes so many classes, functions, and interfaces. All these are used to make and authenticate the managed list of the certificates. These certificates are submitted as certification paths. The certification authority promises the digitized identification with the help of signing the certificates with the Certification Authority’s personal key. If that is not available, then there is a need of a certificate chain and its path (certification). The every certificate promises for the preceding one. The dependent party must have a certificate that is trusted one. On the basis of validated results, the consumers can combine a public key with the subject. The pluggable provider architectures is also provide by the Java CertPath API.

**XML Digital Signature API**

The XML is the acronym for Extended Markup Language that is used to create web pages. There is always a need for securing the web pages. As the guidelines given by the World Wide Web Consortium (W3C), the XML security is required. For this the Java language provides the best API for secure web pages that is XML Digital Signature API. The standard digital encryption services are provided by the XML DS API.

The encryption of the XML is utilized for performing the fine grained fragments, element-oriented fragments to encrypt XML documents. It also encrypts the binary data and that is included within the XML document.
2.1.4 Java Key and Certificate Management

The keys and the certificates are managed in the proper way in the Java language. It is an important functionality in the Public Key Infrastructure (PKI) schemes of the language. There is a particular class available for this purpose that is named as java.security.KeyStore. The class encases all the concept of a key store. It has the cryptographic keys and all the public key certificates. The key store discriminated in three types of entries:

1. Private key,
2. Symmetric key, and
3. Trusted Certificates

The private keys are put jointly with the regarding certificate chain. The keys are stored in the encrypted form. All the trusted entries are intended to hold the certificates which relates to the entities we have authenticated. The functionality of storing the entries is provided by the key store with the help of store method (). This method have output streams, and char [] password. There is no particular form for storing the keys that has been left on the programmer’s decision. The storage media is also a choice of the programmer.

The key store is well protected with the help of the passwords. We can use the other password optionally when we want to apply the safeguards for private and public keys in the store. The public and the private keys are cooperated. This is critical one for fatality of using the Public Key Infrastructure services and protected storage. All these are so much vital to the users.

The PKI is completely based on the cryptography and it provides confidentiality, integrity, and authentication. There are various characteristics of the Java platform which offers a secure computing environment for the programmers.

2.2 Literature Survey

In this chapter all the technologies which are used in the development of the Java File Security System (JFSS) are discussed like Java technology and Rijndael algorithm. And the previous efforts done by the researchers are also considered here. In the section first Java technology is discussed in detail and in the second section the Rijndael algorithm is discussed and in the third one that is the important section in which previous cryptographic/ encryption file systems are discussed.
There has been lot of developments on the thought to use from enciphering to data that is stored in file system. Too much instances are available in the current environment of daily use. Currently one has more than a few cryptographic systems obtainable, some of that have been elaborated here briefly. These include:

2.2.1 CFS

It introduced comparatively resistant enciphering memory system for “UNIX” OS. This file system has been developed by scientist Blaze in 1993.

It furnished a see-through UNIX crossing point to index ladder which mechanically encipher by means of customer furnished keys. CFS utilizes the Data Encryption Standard (DES) cryptographic algorithm to provide security to the data and Meta data of the user files. It is a configuration independent system. It does customer-level cryptography on Network File System (NFS). The NFS loop back device is utilize here to seize the system calls and forward them to the kernel. CFS applies encryption at the granularity of the directory both on the local and remote file system. Files saved in the directory are saved under a diverse mount point and user-attached directories. Consumer must create an encoded index and divides main need in favor of enciphered changes to whenever index happens to be generated very first.

The CFS files are enciphered with the help of sequence of abuser level programs. The commands “mkdir” and “cmkdir” are utilized to generate the directories which are the encrypted one. At the time of directory creation, the abusers enter the key at the appointed time. For utilizing the abuser directories, these are brought to the normal directories first. It always needs the key, the enciphered directory, and the mounted directory name at the time of attachment process.

Diverse cryptographic algorithms are utilized to encrypt the data in CFS, together with DES as talked about in the Data Encryption Standard. The implementation talked in the journalism elaborates how DES with a 56-bit key is utilized in diverse modes to give safety. The 56-bit Data Encryption Standard does not offer sufficient safety. Due to the 56 bit key size, it is prone to the brute force attack as discussed in the literatures. The CFS regulates the DES block cipher. It uses the modes Output Feedback and the Electronic Codebook for generating the key streams from the users passwords.
For using the encrypted directory, the CFS demands the abuser for combining enciphered particular directory named as “/crypt”. The attachment primarily generates the relationship between the mount point directory “/crypt” and the enciphered directory. In this manner the true enciphered matter exists inside enciphered main place, as well as relationship gives windowpane for retrieving such enciphered data within the plaintext from authorized users. DES is used for the encryption of the data in the CFS.

All the file data and the metadata are encrypted by the CFS. The resultant of the encryption of the file metadata is in path names and file names which is 50% bigger than corresponding to the decrypted data. It decreases the size of the legal file names used by the users. The multiple directories with diverse keys and diverse encrypting algorithms per directory are maintained by the CFS. There is extra context switching is done which decreases the performance of the CFS. The basic disadvantage of the CFS is that it is works in the user mode which needs a lot of context switches and copies of the data from user space to kernel space.

2.2.2 MSDOS Safe File System (SFS)

The SFS (an MSDOS device driver), which encodes the complete partition. After encoding, the driver gives a deciphered sight regarding enciphered matter. It furnishes favorable representation regarding file handling component of the OS.

The SFS carries out a encrypted file system for MS-DOS4. Though the conversation takes place around SFS infrequently in Usenet, entire development has been stopped. The documentation of the SFS describes that thirty five passes help to protect the escape of the decrypted data. Over to be, to export freely of academic property dilemmas and restrictions, SFS shuns algorithms and was developed exterior of the USA.

No source code has been relinquished by Gutmann. This is due to the reason that the code can be copied by other encrypted file systems. It is activated only on the only machine. The abusers can only belief on the local operating system and the h/w. There is not the concept of the super users exists on the MS-DOS beginning versions. The unauthorized users cannot take the physical accessing of the files after they have been encrypted. It is not clear that the file metadata is protected or not by the SFS against the unauthorized modifications. The protection is done on the disk partition basis.
Mainly the significant point of the SFS is access methodology at the time of emergency. It can recover the lost passwords, at the data loss time.

The Cipher Feedback (CFB) cipher mode is used for the enciphering process. It is proposed by Gutmaan himself. An intermediate key is generated with the help of password by recursive one way hash function for the password many 100’s of times. The in between key is utilized for decrypting the super input that is used for sequential encipher or decipher the main data on the hard disk.

The SFS documentation appreciates the SFS with reasonable attainments and excellent storage practice. In the common four hundreds and eighty six desk systems, there is very less requirement of RAM by the SFS. The RAM size needed is 1.5 KB. Guttmann says that the SFS is not bound with the single operating system; it can be run on many operating systems.

The two time barrier methodology is given by the SFS to comeback from the losses. The life of a password is limited. After some time SFS needs the new password for protecting the super key. It cannot give safety to the direct attacks to the super key.

So many passwords changes always make the losses lesser ones for the super key. SFS cannot simply portable. At the time of the Windows 95, there is always needed of extra hard works. SFS was not so good to handle the UNIX system’s problems of security.

Figure 2.6: SFS interfaces to MS-DOS to the physical disk.

2.2.3 TCFS
Transparent Cryptographic file system (TCFS) was developed in order to remove the drawbacks of the CFS. It means it is also at cryptographic network file system. Numerous scholars and faculty members of the University at Salerno in Italy, has developed the TCFS in the period 1991. The novel versions of it come regularly with some modifications. The network was used very securely by the use of the TCFS, the secure network file system.

It was the improved security model of the CFS. Confidentiality and the authentications are achieved by the FIPS DES algorithm. It is improved NFS server at the client side to communicate with the distant NFS server. It provides the Remote Procedure Call (RPC) based server. It was provided for the Linux operating system only. It needs both the client and the server on the Linux system.

It behaves like an abstraction level below the Virtual File System (VFS). It makes it totally the transparent to the users of the file system for their applications. The TCFS removes the demands for the “attach” and the “detach” commands which were needed in the previous CFS file system. It permits to set the encrypted properties which are attached with the files and the directories. The UNIX operating system’s authentication is utilized for the authentication of the abusers. All the secure keys are saved in the particular directory. It is the main drawback of the file system which stores the secure keys on the disks and based on the passwords of the users. The TCFS uses the same methodology.

All the secure keys are handled transparently by the TCFS. Here the users are not motivated to give the keys with the help of the specific programs. All the keys are created arbitrarily by the file system. Abusers identifications are utilized as the indices in the key file for accessing the keys for the user files. This transparency is the benefit for the user not to bother about the keys for their files to encrypt or decrypt their data. But it minimizes the security also. The TCFS uses the dictionary password programs for breaking the user passwords with few problems. The password breaking softwares are freeware programs on the internet.

The Remote Procedure Calls (RPCs) are well treated by the TCFS throughout the kernel of the operating system. The read, write RPCs are expended for achieving the security operations. At the every new moment a novel file handler is generated, the expended property “secure” is examined. Every read and write operation for the safe files are filtered from the particular layer named as “Encryption or Decryption layer”. The file system keys are attached
with each user to the encryption or the decryption processes. The passwords are used to encrypt or decrypt the secure keys which are stored in the database “/etc/tcfs pwd”. It is the basic drawback of the file system which uses the user passwords for the keys encryption or decryptions.

The CFS does not encrypt the metadata. So a little confidentiality is lost the file sizes and their names and the subsisting utility applications like fsck for the disk recovery will continue to be operated as usual. There is no documentation regarding the protection from the unnecessary updating by unauthorized users. The standard passwords of the users are used to encrypt the user keys. There is no protection mechanism available here for the data loss cases. The TCFS utilizes the IDEA, DES, and RC5 algorithms for encryption as well as decryption processes at compile time. Global key file is maintained by the TCFS which is usually accessed by the users with their passwords. In the current version of the file system a group secret sharing protocol is utilized. The TCFS file system is so much improved now.

At the same time so many users can log at system for opening particular file within the TCFS file handling component of an OS. A complete set of group users must log on to the working areas. This is not the concept of UNIX operating system of sharing the resources.

The file system is not the robust in the working but the documentations make it powerful as the comparison with the CFS. With the help of the old keys we can access the key generation programs easily. The system calls are used by the TCFS. The set user identification (SUID) is the system call. The root program for the same has so many permissions regarding the type of the program is going to be executed in the system.

The TCFS system needs the binary changes for the login files. This file system does not operate well with the other systems of security like Kerberos or the one key password. The authenticated systems are easily available for the working environment. There are the regular versions for the TCFS file system. It is good example of the cryptography. It makes the file system the robust file system because the developments are continuous for the same.

Here database is utilized for storing the keys for the groups as well as the users. This database is the main drawback for the file system because it can be easily attacked by the eavesdroppers and the intruders. It was basically developed for the distributed operating systems
not for the standalone operating systems. The performance is not acceptable in the case of desktop systems.

2.2.4 MS EFS for Windows

It is utilized by the users of Microsoft based OS. This happens to be the certified file handling component of an OS from Microsoft. It is basically based on the Access Control Lists (ACLs), when the intruder has the physical possession of the computer then the intruders cannot access the files which are encrypted ones by the Windows EFS.

All the enciphering inputs happen to be saved at secondary storage device inside lock-box transparently. The lock-box is locked with the help of user’s passwords. The file system has a big drawback that when the users are going to change the system passwords then the user is required to decrypt the secure files first and then the change of the system password. If the system super user changes the system password then the encrypted files are not accessible by the owner of the files.

The EFS file system utilizes the New Technology File System (NTFS) files for the controls on the files. The files stay encrypted in the buffers of the particular systems. All the files and the folders are specially marked when they are in the position of the encryption. The encryption attribute is used to mark the files and the folders. A hybrid methodology is utilized by the file system with the arbitrary individual file keys.

But these can also be saved on the smart cards of the users for the betterment of security on the new Microsoft Windows Vista operating system. A smart card module is authenticated in this operating system. The default recovery system is used by the system administrator if required at the time of losses. The powerful recovery mechanism is provided by the OS.

EFS is bound in the company of only OS that is Microsoft’s Windows. It provides a lot of better design properties. It does not give the physical access protection. The external setting properties decrease the transparency of the file system. It becomes the problem for the naïve users. They are not well knowledge of the computers so they are not aware about the security of their resources. Even the well acquainted users can also loss their important files if they have little carelessness. Another drawback is that it prone to the physical access boot attacks. Other users can delete the encrypted files of the other users, that is a great loss.
2.2.5 Cryptfs

It is the cryptographic file system which is the robust file system. It is well known file system with its counterparts like eCryptfs and NCryptfs. It encrypts at the level of the user files. So we can say it is file granular encryption file system. In it many files can use the same encryption key. The Blowfish encryption algorithm is used by the Cryptfs file system. The algorithm was in the race of AES, but it fails in the race because its implementation was not easy, so it was not selected as the AES. But it is more secure as compared with the Rijndel algorithm which has won the race of AES. The CBC cipher mode is used for the encryption process. It is the kernel resident file system it means it is more efficient from the others which has been developed inside customer oriented location of the OS. It is mounted at the other underlying file systems.

The Blowfish cryptography is the symmetric key cryptography. The complete file and its metadata is encrypted symmetrically. A one hundred and twenty two bits key is utilized in favor of enciphering as well as deciphering process concerning Cryptfs file system.

The Blowfish algorithm does not increase the size of the files after encrypting the files of any type. So there is no offset value problem in this case. So its performance is very good.

The file system “Cryptfs” does not have any recovery system it is the drawback concerning this. We must be ready for the time of great losses to recover from them. It also encrypts the file names and the path of the files it creates a great confusion among the users. In the UNIX file system; there are so many file names compulsions in the file names which are the valid ones. There are so many special characters are used like forward slash and the null charters which always give the invalid file names. The Cryptfs file system improves such type of problems very easily. It does not encrypt the special directories with single dot and the continuous double dots.

The stackable layers are utilized by the Cryptfs encrypting file system. Its implementations are very fast in the functionalities. Because of this type of stackable environment we can use any number of file systems on a single system according to our needs. There is no confusion among these. The Cryptfs gives a large quantity of kernel codes. There is no need to be worried about the low level coding the file system resolves it very transparently.
The super user of the system can mount the file system at any partition concerning this. It is completely kept trustable with system. Cryptfs efficiency of this is so much advanced by the kernel implementations of the layer. There is not the extra burden for the context switches as in the case of the user space file system.

2.2.6 eCryptfs

This cryptographic file system was developed for the Linux freeware operating system. It is the extended edition related to preceding “Cryptfs”, so name is given after that file system as “eCryptfs”. Erez Zadok et al. have developed this file system. It is also based on per file per key based file system. Every file has its own arbitrary encryption key that is called FEK. It is utilized for encrypting or decrypting the matter within the concerned files. These keys are also saved on the secondary storage device as is the case of the previous file systems except the Window’s Vista’s EFS.

It is also stackable file system implementations as the case of the Cryptfs file system. The metadata of the file is also encrypted here but that is saved in the header of the files. The sparse files are not compatible regarding the storage system formats. It is also based on per file encryption or decryption key. This key is encrypted with the help of the session wide keys which are given from side to side a pass phrase. These are the authentication tokens per session. It means that if there is diverse authentication then we need to remount. It acts as a stackable layer above the basic file system. The stackable layer for the eCryptfs has key management operations as well as encryption or decryption operations to be done in it. After that it passes the data to the basic file system.

It has the kernel key infrastructure. It gives the PAM (Pluggable Authentication Module). This module is not completely developed. It is a flexible and secure coat.

Here two modes are utilized for the key and the metadata saving. Firstly is very ordinary mode which changes the structure. File header gives encrypted context for the decryption of the data of any user with enough permission to the particular user for accessing the files.
The file has been portioned in the extents. These extents are behaved as the cryptographic part with its individual vector. The initialized vectors (IVs) for every group of extents are saved in one of the previous extent. It follows the set and so on. It is done till the last point of the file.

Though, the administrator is still trusted one. It becomes the drawback of the file system.

2.2.7 BestCrypt File System

BestCrypt is a business-based file system it means it is purchased by the customers. It is a cryptographic file system. It generates a novel chunk machine by single file and the encryption process takes place by the system calls interception. It manages the containers. These containers can be easily generated, mounted and managed on the computer machine. These containers are basically the encrypted volumes. These act as novel mounted partition and saved as the encrypted files when the files are not in utilization. In this manner the important data is protected from the intruders or the unauthorized users.

Since these containers are normal folders, they can be easily transferred to the removable electronic devices like flash devices, and the portable hard disks etc. We can also send these containers through the emails on the networks. These can be easily used in the distributed world. These can be easily mounted on the client computers.

This only file deeds as the promoted pile, on that as a file system arranged will may, and consequently called container. Every container has a connected encryption key that will become in Cryptographic changes used. Containers must be created to be arranged, and to be set up by the overseer. This device can be arranged with an obtainable file system.

This file system compels the users to generate and mount the containers. The marketable, accessible software happens to be fare in favor of consumer surroundings anywhere formation as well as utilization of the container to an individual is limited. In more customer surroundings, consumers must divide the figure that is connected in the company of container and system overseer, while functioning excessively super-customer needs. The BestCrypt file system holds so many best encryption algorithms like the AES (Advanced Encryption Standard), the Rijndael algorithm which is the United States standard. It also has to more known algorithms which were in the race of the AES selection. These are the Ron Rivest’s RC6, and the Twofish algorithms. Triple DES is also provided by the file system. The Blowfish is used with many variations of the
keys sizes. Other algorithms are like IDEA, CAST, and the GOST. Simple DES is also supported by the BestCrypt. But it is not recommended by the vendor.

At the termination, BestCrypt is an inspiring invention. The firmness, inclusive and modular sustain for a diversity of well-liked encryption algorithms, and generally stiffness are something to go by, it appears to be extremely safe. It is excitedly suggested to suppose the BestCrypt for important files. It can work on the operating systems Linux and the Windows.

2.2.8 Steganographic File System (StegFS)

The steganography and the cryptography, both the methodologies are used in the StegFS file system. It is prepared for the desktop computer machines and the servers which have their local storage devices attached. It is not unconditionally correct to stage unlike local systems, how for example a distributed storage that is overcome against with extra security dangers. The distributed systems require more security than the desktop or standalone systems because they are accessible on the network so any intruder can intrude form any place. We always in the need of defend of the data that is hidden one.

The various layers of security are provided by the StegFS to the users. Every layer has its individual security key. It is a updated EXT2 kernel driver. The block allocations are managed according to the security layers. The opponents always takes an eye on the system, they have the knowledge which has a little unseen or covered matter obtainable. But they are not aware about matter. They are well packed for the security purposes. Data can be duplicated arbitrarily over the disk to comeback from the losses. The arbitrary blocks can be overwritten by the EXT2 driver.

In the StegFS the files are hidden for the security motive. The file system has so many extra properties. But it also has so many drawbacks. These are discussed as follows.

- Every hidden file is saved at the one location. So we cannot choose the individual files from the one location. All these files are stored at a temporary block. From this place the user’s can take the needed files above the StegFS Volume.
- There is fragmentation problem with the file system so many temporary files are stored. It is used for the efficiency purpose of the users but it becomes the limitation.
The expired user session account’s hidden files are not removed, so the users who have the concerned file’s private keys can access the data.

Figure 2.7: The StegFS kernel driver

Though the StegFS reaches physically powerful contestability of data the survival, is the achievement dismantling very high, building it unreasonable for the majority of applications.

### 2.2.9 TransCrypt

TransCrypt belief on very few number of units, defends opposite to broader challenge structure, furnishes finely file part as well as uprightness shielding. It uses one kernel place sew on wrestled for larger safety as well as good quality achievement. It provides decisive honor between the kernel and the consumer the place of a security viewpoint.

It unites supports a higher key direction scheme, that excludes the overseer account and approved consumer of the belief model and, the use of hardware certification sign. The extra claims by business environments such as for example safe division distant, data recovery and immunity by attacks, that by the inside shot down will be encountered also. Encouraging initial achievement estimations show of this safety as well as working advantages may be utilized in the company of tiny as well as the operation costs by tolerable running time.

### 2.2.10 NCryptfs
NCryptfs recovers take action substantially on Cryptfs specifically in the furnishing arranged up, and exceeding the session specific certification scheme.

NCryptfs is certain a file system that is created to furnish comfort, and high achievement. It is also a stackable file system like its counterparts Cryptfs and eCryptfs. It can be piled on the existing file systems which come with the operating systems.

The super users of the systems have faith for suitable mounting points and the user based units. It has no trust on the provided security keys. Every user has been differentiated with each other in this file system. Here the users decide the rules for the other users of the system. In it the mount feature is called as the “attach” in the user mode. Session based authentication is applied here. The keys are to be brought on the command line prompt if its policy gives permission for the utilization.

Here the attachments are permitted to the users. It is the same methodology which is used in the Cryptographic File System of Matt Blaze. Here the file sharing is done very easily by the users. The attachments can be shared with the other group members. These members have the great knowledge of the secret keys. This sharing makes the system more comfortable to the users. And the users behave like the owner of the files.

The UNIX operating system grouping is also maintained with the help of NCryptfs. A novel kind regarding cipher mode is utilized here. This cipher takes an input from the random size buffer and then it is encrypted to a set of size. The encrypted files are also saved on the file system. But all the files are saved with the hashes which are derived from the absolute names of the attacks.

So many attractive properties are owned by the NCryptfs file system. But the directory structures are not secured by the file system. It remains as it is. It is prone to the attacks. The file system is not convenient to the users because it utilizes the command system for encrypting the files. The command is used named as “attach”. The user based process are also become reliance for the file system. It is very prone to the intruder attacks. Also the transparency is not provided by the file system.

2.2.11 EncFS
EncFS is also a user based file system. A pass through plan is utilized by the system. Its model is on the basis of the most prominent encryption file system named as “CFS” which is invented by the scientist Matt Blaze in nineteen hundred and ninety three. The FUSE library is utilized here. The FUSE stands for the File system in User Space. One Linux based kernel module is used for the interfacing of the FS. This happens to be cryptographic FS. This takes files as an input not the folders at a time. The complete block device is not considered as an input by the file system. The pass phrases can be modified by the users at any time. The EncFS can be utilized on any underlying file system. There is no need to change the partitions of the secondary storage devices or the file systems.

EncFS file system is a freeware. The transparency is very well achieved by the file system. All the encrypted files are stored on the random directory by the file system. But the other file systems use the fixed space for the encrypted files for storage. It does not reserve the big amount of secondary storage space at the very starting stage of the file system. We can easily make the file directories on the underlying file system. At these directories we can easily store the encrypted files. The file system is easy to install and use. There are so many encryption algorithms are available for securing the data on the files. These are the Blowfish, AES (Advanced Encryption Standard). There may be varied cipher key length which is selected for the ciphers.

The file system can be easily backed up on other locations. It has a backup system which can detect any change within the files of the users. And it is helpless to decipher such type of files. The backups can be generated very easily without the need of any mount point. All the files are saved independently as enciphered files. The directory tree is used for the encrypted files.

The EncFS file system has so many limitations with it. These are discussed here as follows:

- Metadata is not encrypted, it is visible to anybody. The metadata contains the information like number of encrypted files, file permissions, files size etc.
- The volumes of the file system are not formatted with random file systems. All the features and compulsions are shared by them on the source directory.
- Fragmentation is another problem with this system.

2.2.12 CryptDisk
CryptDisk cryptographic file system is used at the Macintosh operating system. Confidentiality is assured to the complete partition. It is a shareware product that was developed in year nineteen hundred and ninety five. The inventor was Will Price. He was the student at the University of Southern California. Now, this encryption file system has become a business item. Its new name is PGPdisk.

On the secondary storage device a container is managed which acts like a virtual disk. It is simply mounted on a partition. The file to which we want to secure, we have to drag it on the icon of the container application of the CryptDisk file system. It is the completely transparent system while we use the dragging and dropping properties of the operating system. It has a strong recovery system. It makes the file system more reasonable to the users. It works like the Secure File System (SFS) in the case of the unmounting of the partitions. The encrypted files are utilized by the users as they are the ordinary files.

It is well trusted file system like the CFS. The multiple users have nothing and the basic account in the Macintosh system as it was present in the DOS based file system. The file system depends on the locality of the operating systems and the local hardware. There is no networking concept available to the file system.

The IDEA algorithm is utilized by the file system for encrypting and the decrypting of the files. It uses the CFB (Cipher Feedback) cipher mode of encryption and decryption. The initialization vector (IV) is used and it is randomized after the each five hundred and twelve bytes of data.

The file system utilizes the function which drives the master key from the user password. The password length may be of maximum one hundred and twenty eight characters long. It is hashed to a random number of moments. It becomes salty with arbitrary data for providing a best encryption key. Session keys are also utilized to secure the own files. The session-based super key is utilized to generate a user session secret key regarding file as well as a well understood arbitrary salty value. This salt always differs for every file. It is very simple to alter the passwords.

2.2.13 SFS
It was generated as a protected substitute for NFS. Here does not exit a key centric server based system for sharing the files on networks. The public key is used through the pathnames to share the files. It permits the transparency in the encrypting file system. There are a few intrusions regarding administers. It utilizes the algorithms like SHA – 1, RC 4, Blowfish for the encryption of the files. There were three main motives behind the development of SFS. These are as follows:

- **Security:** It is assumed that the intruders have a complete control over the network. That control is a great hindrance inside FS operations on computer networks. FS obscures the communication takes place on the network.

- **Global Namespace:** All the distant file systems are mounted below the directory /sfs. The directory’s contents are same on each client in the worldwide.

- **De-centralized control:** The file system has not some licensed power to maintain the global namespaces. The file servers can be established with a computer which is connected to the internet. There is no need to be get certifications. Every client can use the novel servers immediately in the globe.

**Summary**

The majority of the file systems preserve the cashing of the pages in the mode of encryption. But, it offers simpler developments. The encryption and decryption in the buffer cache is quicker one. But, it is very prone of memory assaults to sustain as page cache. Except the Microsoft EFS, there is no cryptographic file system which utilizes the mixed approach of symmetric key and asymmetric keys.

In all the file system implementations, the administrator is completely taken in the confidence except the NCryptfs. The minimum confidence is taken on the administrator in the consideration in the file system TransCrypt. Each file system containing NCryptf, all have disregarded the attacks from masquerading. In this type of attack the user space facilities can become false. These types of attacks are compacted within the TransCrypt. In this file system, for the access control and authentication, it offers a unique key for every consumer. It provides new proposal for difficulties in administrator oriented attacks. It provides the one system based key. Many of the designs offer the transparency to the users. In the TransCrypt file system all the file operations are completely transparent to the users.