# REMOTE CLIENT AUTHENTICATION

**P.SWETHA ***

**R.ARUN KUMAR****

***ABSTRACT:***

This system is developed on relation towards today's security challenges which shows the effectiveness of remote client authentication schemes varies extensively, like which include various attacks like phishing, man in the middle and other malicious software. By developing this remote authentication methods shows how each measures up and it also includes recommendations for solution developers and also consumers. So by providing a security can however difference with business or usability goals. So it might be acceptable. So by this developers are more concerned with cost and minimal user training and support than with the treat of improper authentication.

***Keywords: EMBEDDED C, REMOTE, SECURITY, AUTHENTICATION***

* Assistant Professor, CSE, Global Institute of Engineering & Technology, Hyderabad

** Assistant Professor, CSE, Vidya Jyothi Institute of Technology, Hyderabad.

## I. INTRODUCTION

In now a days we know that today's security challenges like phishing, man in the middle and various malicious software's attacks to provide various harmful data losses and code corruption to the existing data. To overcome from this kind of attacks we need a security means by providing remote authentication is widely used at anywhere. Remote means it provides a way of connection is provided for communication between client and server. One of the good example for this is Internet banking which it comes the added challenge of a user base that's not necessary for technical ability. After that researchers have proposed many remote authentication methods including simple passwords, public-key infrastructures (pkis), biometrics, smart cards, and mobile phones. Each mechanism has a reason to exist on the design criteria. The challenge therefore has become less one of inventing a working scheme, and more of deciding like which scheme to choose given the design criteria. Finally security can conflict with business applications. It might be acceptable why because for example to deploy a password on basing on authentication solutions, so by this developers or more concerned with cost and minimal user training and support than with the threat of improper authentication.

**2. Objective:** The main purpose of the system is to provide java programs to provide a effectiveness of remote authentication methods which is preventing from a various malicious software's attacks.

**2.1 Existing System*:*** The current system is gives a solution from today's security challenges is to use a security device with a display and embedded keypad that maintains a secure end to end connection with the server, by this it protects against various malicious software's attacks.

**2.2 Proposed System:** The proposed system is provide a PKI based remote authentication scheme by this server usually stores either copies of the certificates or corresponding hash values that it can authenticate.

**3. System Analysis:** The first step in developing anything is to state the requirements. This applies just as much to leading edge research as to simple programs and to personal programs, as well as to large team efforts. Being vague about your objective only postpones decisions to a later stage where changes are much more costly.

The problem statement should state what is to be done and not how it is to be done. It should be a statement of needs, not a proposal for a solution. A user manual for the desired system is a good problem statement. The requestor should indicate which features are mandatory and which are optional, to avoid overly constraining design decisions. The requestor should avoid describing system internals, as this restricts implementation flexibility. Performance specifications and protocols for interaction with external systems are legitimate requirements. Software engineering standards, such as modular construction, design for testability, and provision for future extensions, are also proper.

Many problems statements, from individuals, companies, and government agencies, mixture requirements with design decisions. There may sometimes be a compelling reason to require a particular computer or language; there is rarely justification to specify the use of a particular algorithm. The analyst must separate the true requirements from design and implementation decisions disguised as requirements. The analyst should challenge such pseudo requirements, as they restrict flexibility. There may be politics or organizational reasons for the pseudo requirements, but at least the analyst should recognize that these externally imposed design decisions are not essential features of the problem domain.

A problem statement may have more or less detail. A requirement for a conventional product, such as a payroll program or a billing system, may have considerable detail. A requirement for a research effort in a new area may lack many details, but presumably the research has some objective, which should be clearly stated.

Most problem statements are ambiguous, incomplete, or even inconsistent. Some requirements are just plain wrong. Some requirements, although precisely stated, have unpleasant consequences on the system behavior or impose unreasonable implementation costs. Some requirements seem reasonable at first but do not work out as well as the request or thought. The problem statement is just a starting point for understanding the problem, not an immutable document. The purpose of the subsequent analysis is to fully understand the problem and its implications. There is no reasons to expect that a problem statement prepared without a fully analysis will be correct.

The analyst must work with the requestor to refine the requirements so they represent the requestor's true intent. This involves challenging the requirements and probing for missing information. The psychological, organizational, and political considerations of doing this are

beyond the scope of this book, except for the following piece of advice: If you do exactly what the customer asked for, but the result does not meet the customer's real needs, you will probably be blamed anyway.

Modules:

1. client subsystem
2. server subsystem
3. digital certificate subsystem

**Modules Description:** The entire system is decomposed into three subsystems the client subsystem, the server subsystem and the digital certificate subsystems.

**1. client subsystem:** The **client subsystem** is the interface for the user to enter the user name and password that can be validated by the digital certificates that are issued by the third party certification authority.

**2. server subsystem:** The **server subsystem** will handle the profiles that are authenticated. These systems will use the cryptography to store the user profiles. The passwords are stored in the encrypted format.

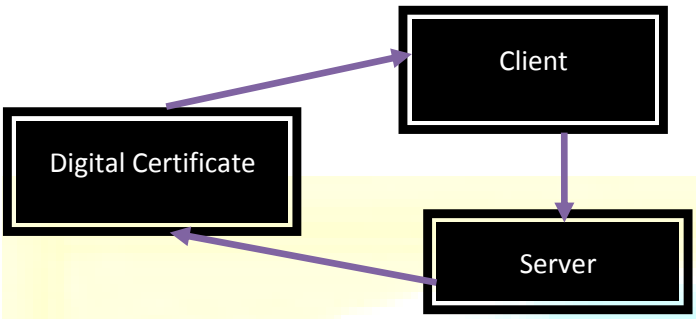
**3. digital certificate subsystem:** The **digital certificate subsystem** is the third party subsystem that will manage the keys for issuing the digital certificate and can handle the digital certificates. These digital certificates generated are issued to the client for the authentication of users.

**4. Design Engineering:** Design is a meaningful engineering representation of something that is to be built. Software design is a process through which the requirements are translated into a representation of the software. Design is the place where quality is fostered in software engineering. Design is the perfect way to accurately translate a customer's requirement in to a finished software product. Design creates a representation or model, provides detail about software data structure, architecture, interfaces and components that are necessary to implement a system.

**4.1 PROPOSED SOFTWARE ARCHITECTURE:**

The proposed Software architecture consists of three tiers. The first tier is the client tier. The client tier will be provided with an interface for creating the client software which uses the remote interface to authenticate the users. The second tier is the certificate management tier. In

this tier the system will manage the digital certificates, some of the services of this tier is issuing the digital certificates and validating digital certificates. The server tier, In this tier the user management will be done. The user management is very complicated.




**4.2 Technology Description:** A programming tool or software tool is a program or application that software developers use to create, debug, maintain, or otherwise support other programs and applications. The term usually refers to relatively simple programs that can be combined together to accomplish a task.

**4.2.1Java Technology:** 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.

### 4.2.2 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.

**4.2.3Java 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.

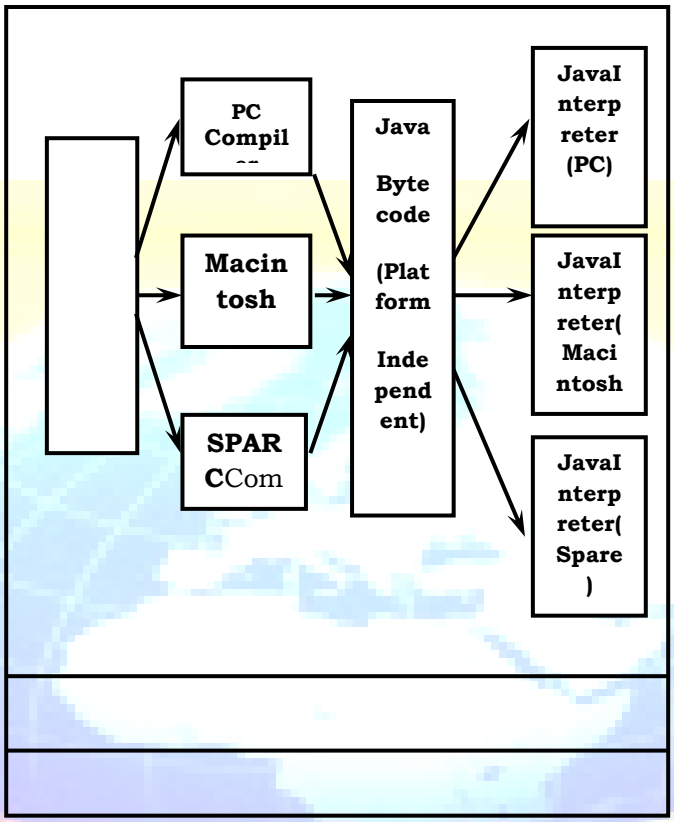
**4.2.4 Features of Java Security:** Every time you that you download a "normal" program you are risking a viral infection. Prior to Java, most users did not download executable programs frequently, and those who did scan them for viruses prior to execution. Most users still worried about the possibility 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.

**4.2.5 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.

**4.2.6 The Byte code:**

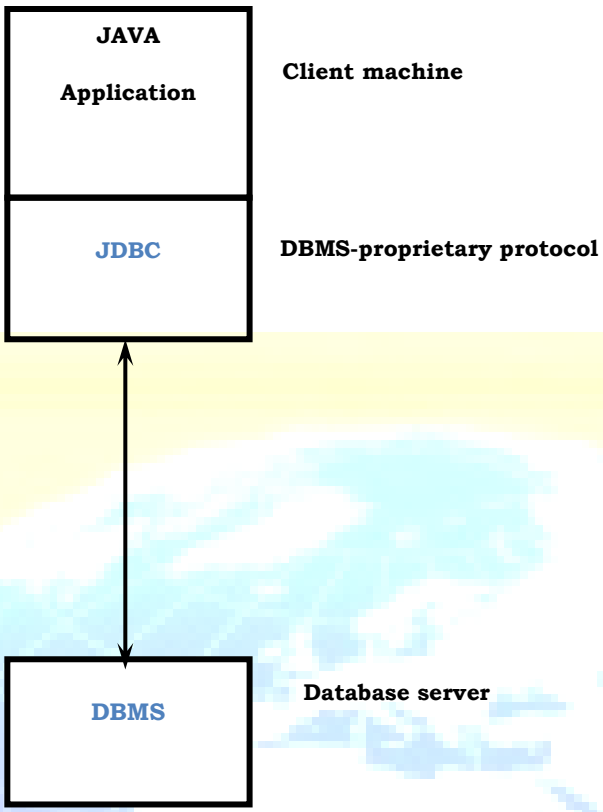
The key that allows the Java to solve the security and portability problems is that the output of Java compiler is Byte code. Byte code is a highly optimized set of instructions designed to be executed by the Java run-time system, which is called the Java Virtual Machine (JVM). That is, in its standard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




**4.2.7Java 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.

4.2.7(a) Compiling and interpreting Java Source Code:



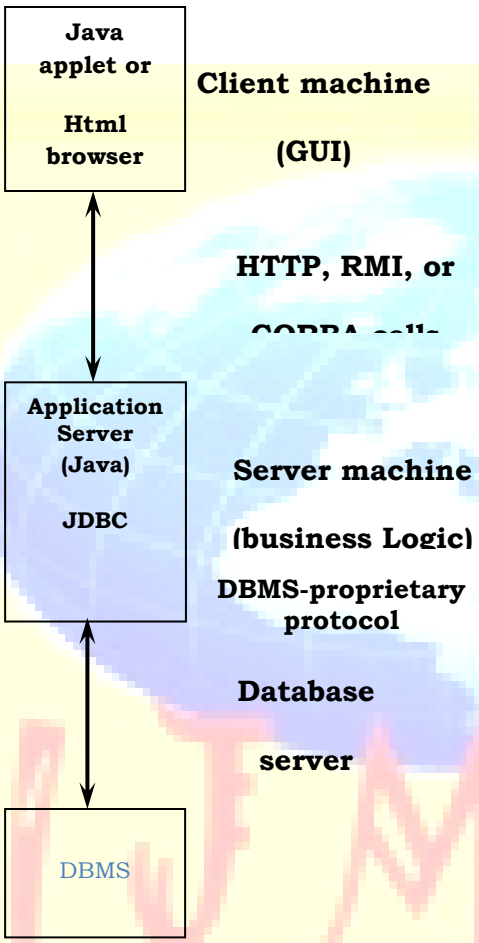

4.2.8 Compilation of code: When you compile the code, the Java compiler creates machine code (called byte code) for a hypothetical machine called Java Virtual Machine (JVM). The JVM is supposed to execute the byte code. The JVM is created for overcoming the issue of portability. The code is written and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

**4.2.9Compiling 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 Suns ARC station running Solaris or Apple Macintosh running system and all could receive code from any computer through Internet and run the Applets.

**4.2.10Simple**: 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

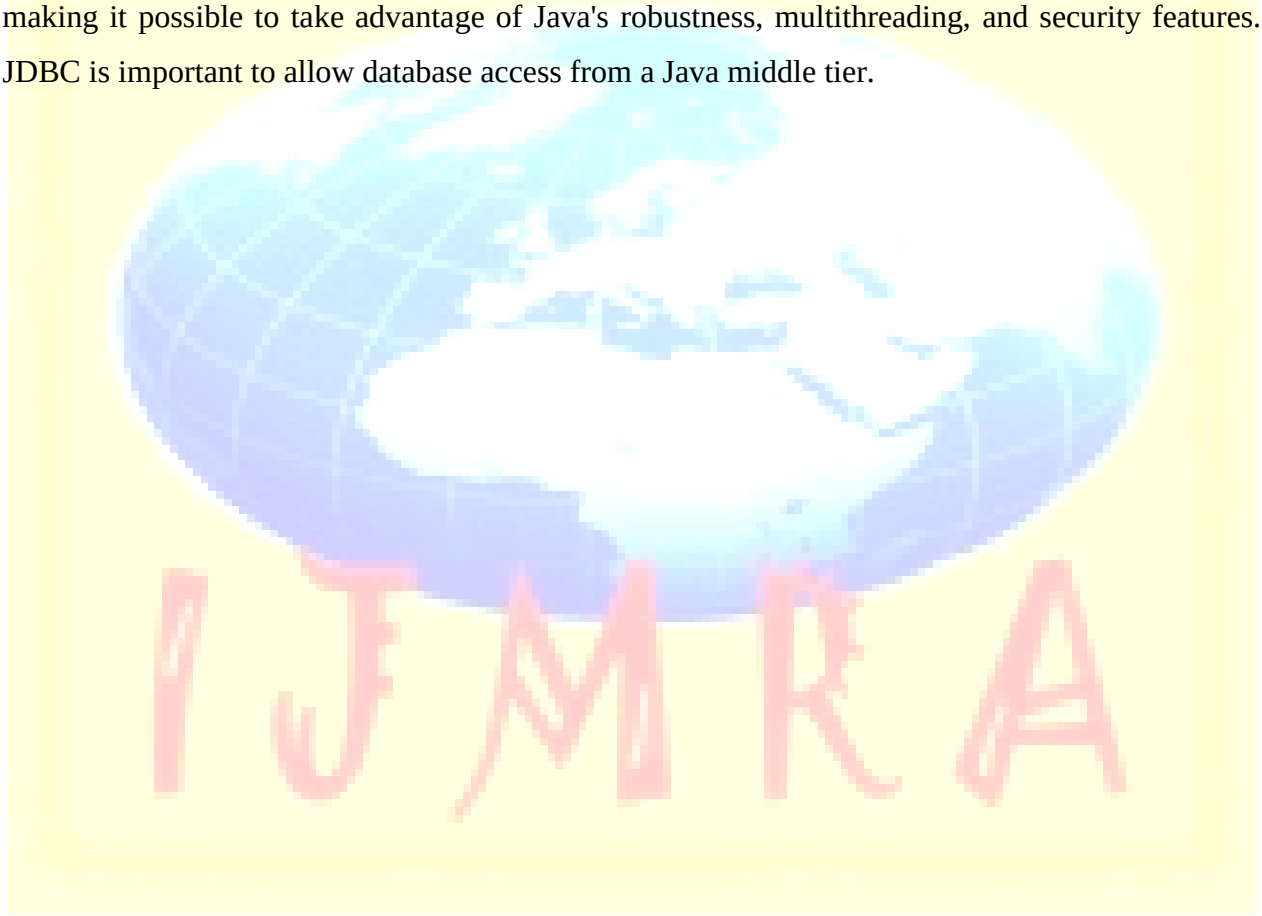
approachable manner. In Java there are a small number of clearly defined ways to accomplish a given task.

**4.3:Two-tier and Three-tier Models:** The JDBC API supports both two-tier and three-tier models for database access. In the two-tier model, a Java applet or application talks directly to the database.




This requires a JDBC driver that can communicate with the particular database management system being accessed. A user's SQL statements are delivered to the database, and the results of those statements are sent back to the user. The database may be located on another machine to which the user is connected via a network. This is referred to as a client/server configuration, with the user's machine as the client, and the machine housing the database as the server. The network can be an Intranet, which, for example, connects employees within a corporation, or it can be the Internet.In the three-tier model, commands are sent to a "middle tier" of services, which then send SQL statements to the database. The database processes the SQL statements and sends the results back to the middle tier, which then sends them to the user. MIS

directors find the three-tier model very attractive because the middle tier makes it possible to maintain control over access and the kinds of updates that can be made to corporate data. Another advantage is that when there is a middle tier, the user can employ an easy-to-use higher-level API which is translated by the middle tier into the appropriate low-level calls. Finally, in many cases the three-tier architecture can provide performance advantages. Until now the middle tier has typically been written in languages such as C or C++, which offer fast performance. However, with the introduction of optimizing compilers that translate Java byte code into efficient machine-specific code, it is becoming practical to implement the middle tier in Java. This is a big plus, making it possible to take advantage of Java's robustness, multithreading, and security features. JDBC is important to allow database access from a Java middle tier.



**4.4 Results:**

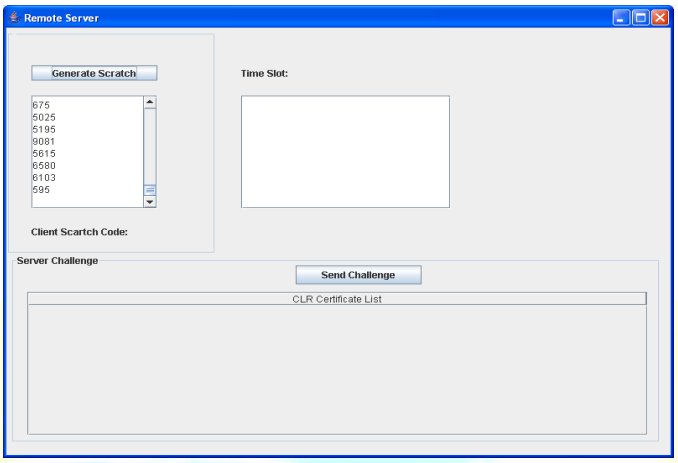



**Fig: 4.4.1**

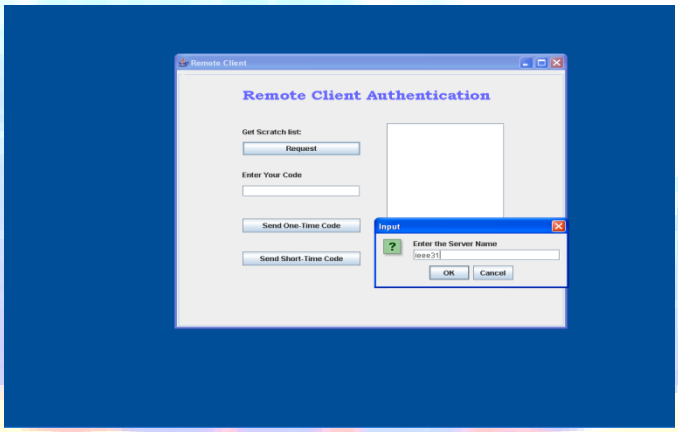



**Fig: 4.4.2**

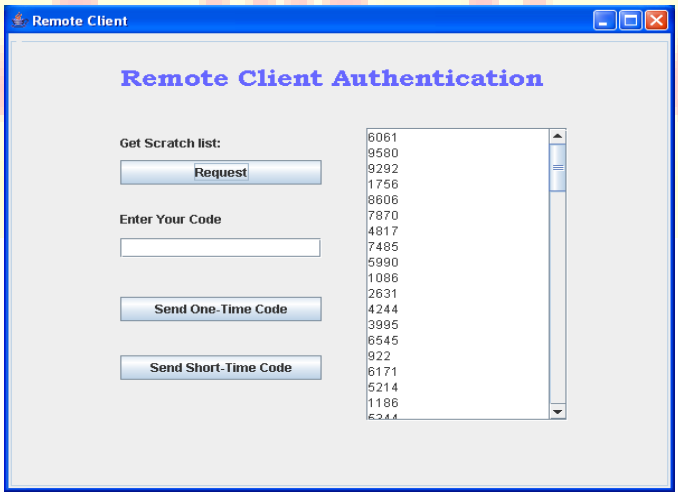



**Fig: 4.4.3**

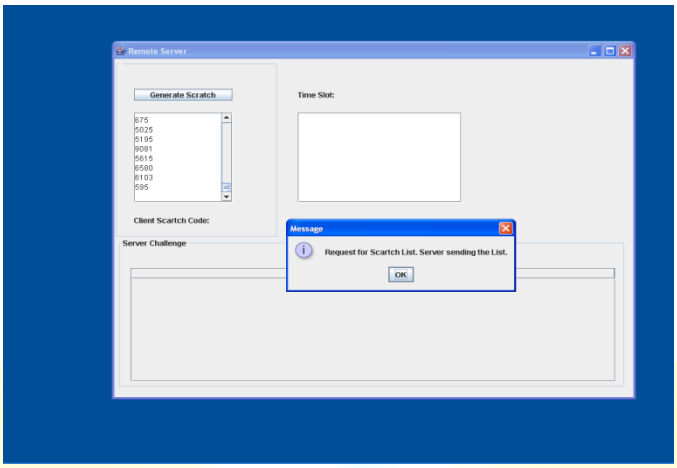



**Fig: 4.4.4**

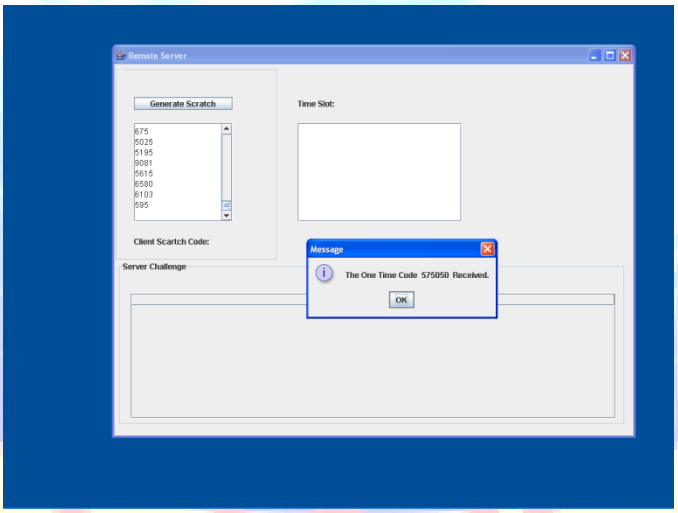



**Fig: 4.4.5**

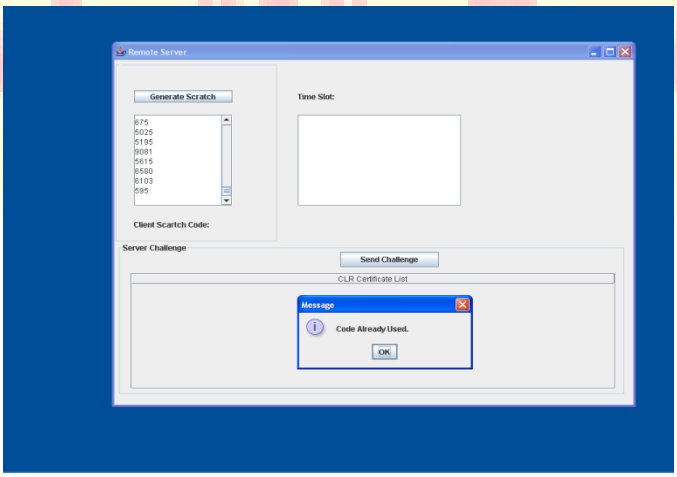



**Fig: 4.4.6**

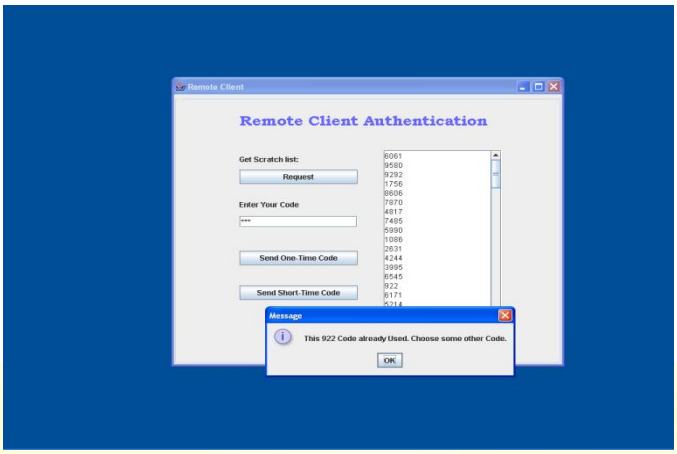



**Fig: 4.4.7**

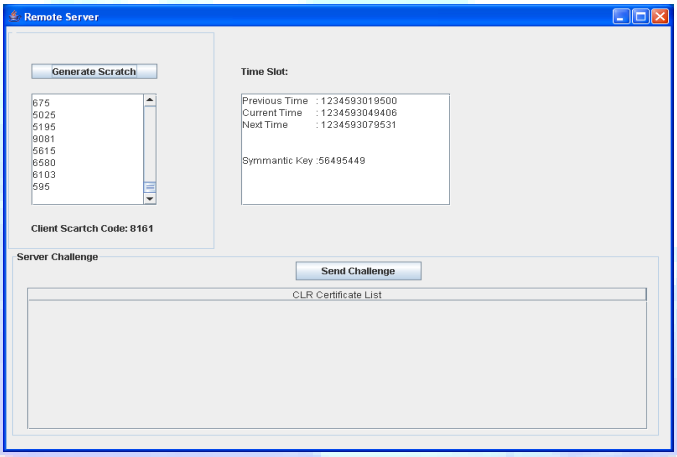



**Fig: 4.4.8**

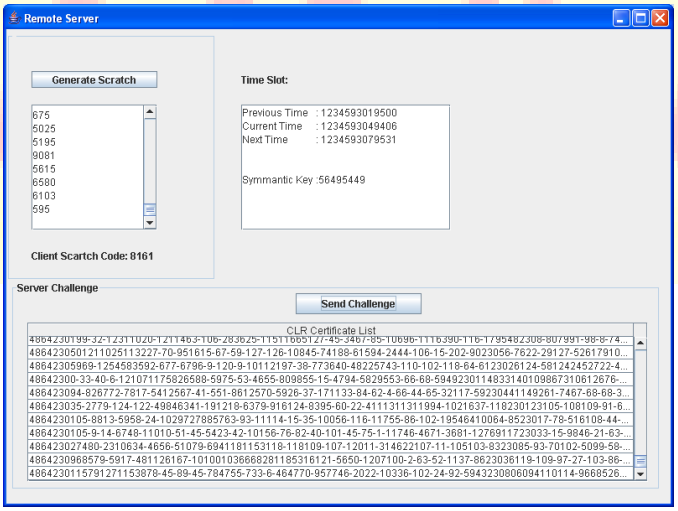



**Fig: 4.4.9**

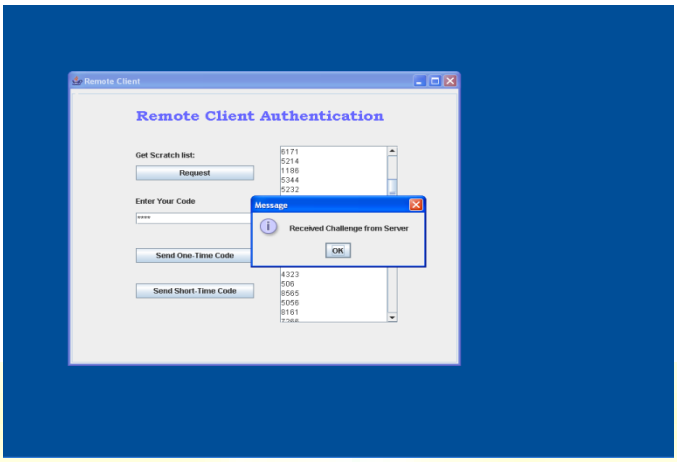



**Fig: 4.4.10**

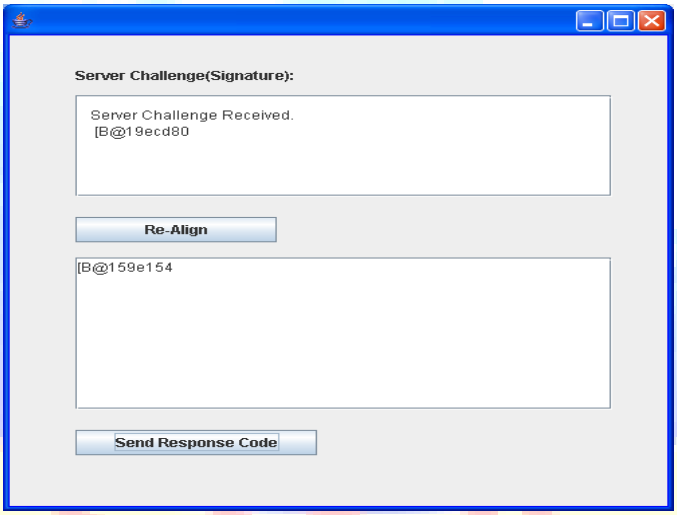



**Fig: 4.4.11**

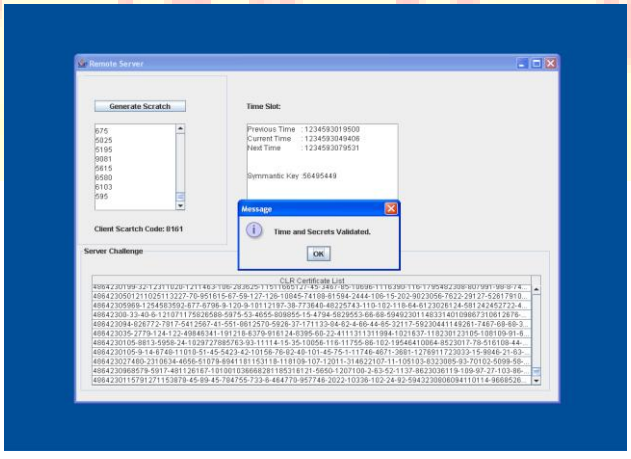



**Fig: 4.4.12**

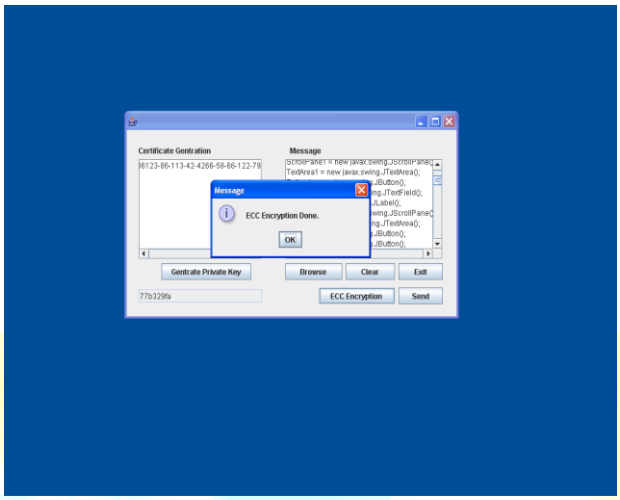



**Fig: 4.4.13**

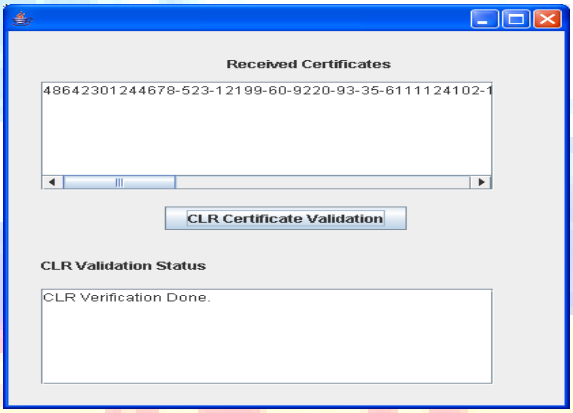



**Fig: 4.4.14**

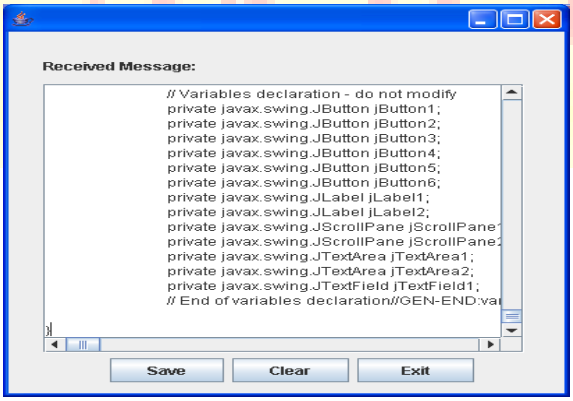



**Fig: 4.4.15**

**5. Conclusion:** we have three primary recommendations for any new remote authentication solution:

- Although still widely used, scratch lists are no longer state of the art as they can't withstand phishing and malicious software attacks.
- Challenge/response one-time codes or PKI-based schemes, combined with a secure device, should be the basis for any authentication solution.
- Because MITM attacks are increasing, 12 developers should build solutions with a clear vision of how they might be extended to thwart MITM attacks.

An ideal solution to counter today's security challenges is to use a security device with a display and embedded keypad that maintains a secure end-to-end connection with the server.13 This protects credentials against phishing, renders MITM attacks impossible, and lets users detect malicious software attacks.

**5. BIBLIOGRAPHY**:


1. B. Schneier, "Two-Factor Authentication: Too Little, Too Late," *Comm. ACM*, vol. 48, no. 4, 2005, p. 136.
2. L. Lamport, "Password Authentication with Insecure Communication," *Comm. ACM*, vol. 24, no. 11, 1981,pp. 770–772.
3. R.E. Smith, *Authentication: From Passwords to Public Keys*, Addison-Wesley, 2002.
4. U. Waldmann et al., "Protected Transmission of Biometric User Authentication Data for Oncard- Matching," *Proc. ACM Symp. Applied Computing*, ACM Press, 2004, pp. 425–430.
5. X. Leroy, "Java Bytecode Verification: Algorithms and Formalizations," *J. Automated Reasoning*, vol. 30, nos. 3-4, 2003, pp. 235–26

**6. Authors Biography:**

P.Swetha is from HYDERABAD, ANDHRAPRADESH. Completed M.TECH in CSE with specialization CSE from Vidya Vikas Institute of Technology affiliated to JNTUH in 2009 and B.TECH in CSE from VidyaJyothi Institute of Technology affiliated to JNTUH in 2006. Currently she is working as an Assistant professor in CSE department at Global Institute of Engineering & Technology, Hyderabad from 2010. Her areas of research interests include Data Mining, Networking, Android & Network security.

ARUN KUMAR.R Completed M.TECH in CSE with specialization (COMPUTER SCIENCE ENGINEERING) from CVSR college of engineering affiliated by JNTUH in 2012. Currently he is working as an Assistant professor in CSE department at Vidya Jyothi Institute of Technology, Hyderabad. he is having more than 4 years experience as an assistant professor. His areas of research interests include NETWORKS, ANALYSIS OF ALGORITHM, and COMPILER AND LANGUAGE PROCESSING.