IOSR Journal of Engineering (IOSR JEN) **www.iosrjen.org**
ISSN (e): 2250-3021, ISSN (p): 2278-8719
PP 30-35

OTPAAS—One Time Password As A Service Alternative Title: Multiple Service Authentications With Cloud OTP As A Service.

Mr B.Anbarasu(ME), Anusuya S
Department of CSE, Sri Venkateswara College Of Engineering Technology, Thiruvallur – 602 024.
Reg.No: 112417405001 Department of CSE, Sri Venkateswara College Of Engineering Technology, Thiruvallur – 602 024.

Abstract: *Conventional password-based authentication is considered inadequate by users as many online services started to affect each other. Online credentials are used to recover other credentials and complex attacks are directed to the weakest one of many of these online credentials. As researchers are looking for new authentication techniques, one time passwords, which is a two-factor authentication scheme, looks like a natural enhancement over conventional username/password schemes. The manuscript places the OTP verifier to the cloud to ease adoption of its usage by cloud service providers. When the OTP verifier is placed on the cloud as a service, other cloud service providers could outsource their OTP deployments as well as cloud users could activate their respective account on the OTP provider on several cloud services. This enables them to use several cloud services without the difficulty of managing several OTP accounts for each cloud service. On the other hand, OTP service provision saves inexperienced small to medium enterprises from spending extra costs for OTP provisioning hardware, software, and employers. The paper outlines architecture to build a secure, privacy-friendly, and sound OTP provider in the cloud to outsource the second factor of authentication. Cloud user registration to OTP provider, service provider activation, and authentication phases are inspected. The security and privacy considerations of the proposed architecture are defined and analyzed. Attacks from outsiders, unlink ability properties of user profiles, attacks from curious service providers or OTP verifiers are mitigated within the given assumptions. The proposed solution, which locates the OTP provider in the cloud, is rendered robust and sound as a result of the analysis.*

I. Introduction

Aim:

The main aim of this project is to build a secure, privacy-friendly, and sound OTP provider in the cloud to outsource the second factor of authentication.The requirements specification is a technical specification of requirements for the software products. It is the first step in the requirements analysis process it lists the requirements of a particular software system including functional, performance and security requirements. The requirements also provide usage scenarios from a user, an operational and an administrative perspective. The purpose of software requirements specification is to provide a detailed overview of the software project, its parameters and goals. This describes the project target audience and its user interface, hardware and software requirements. It defines how the client, team and audience see the project and its functionality.

II. Proposed System

The proposed system lets companies spend less on OTP-based TFA transition both in the perspectives of experience, employers, hardware and software. Additionally, it lets the users to manage many of their accounts easily at one place, yet via unlikable profiles. It is believed that outsourcing OTP service in the cloud may also ease many cloud service providers bulk OTP adoption, as they do not require making additional investment.

The proposed approach is effective as a two factor authentication security mechanism and provides many configurable options by design. User profiles are open to future development at user devices, such as regular password management, credential management etc.

The Java Programming Language:

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

- Simple
- Object-oriented
- Distributed
- Interpreted
- Robust

- Secure
- Architecture-neutral
- Portable
- High-performance
- Multithreaded
- Dynamic

The code and can bring about changes whenever felt necessary. Some of the standard needed to achieve the above-mentioned objectives are as follows:

Java is unusual in that each Java program is both co implied and interpreted. With a compiler, you translate a Java program into an intermediate language called **Java byte codes** – the platform independent codes interpreted by the Java interpreter. With an interpreter, each Java byte code instruction is parsed and run on the computer. Compilation happens just once; interpretation occurs each time the program is executed. This figure illustrates how it works:

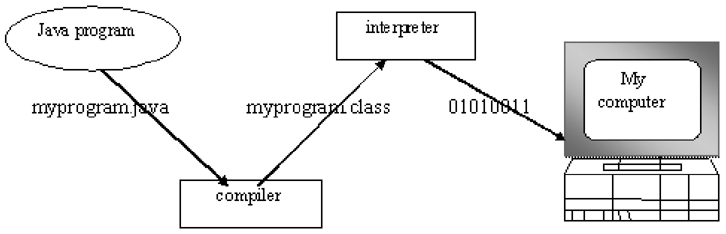
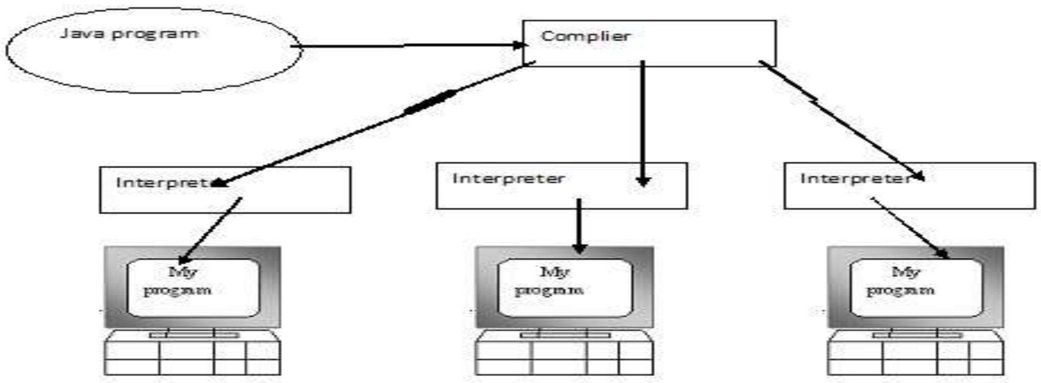


Fig.3.1

You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (JVM). Every Java interpreter, whether it's a Java development tool or a Web browser that can run Java applets, is an implementation of JVM. That JVM can also be implemented in hardware. Java byte codes help make "write once, run anywhere" possible. You can compile your Java program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the JVM. For example, that same Java program can e run on Windows NT, Solaris and Macintos.



Windows NT **System 8**

d) The Java Platform:

A platform is the hardware or software environment in which a program runs. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other, hardware-based platforms. Most other platforms are described as a combination of hardware and operating system.

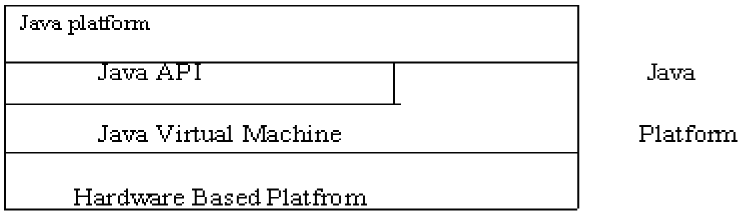
The Java platform has two components:

➢ The Java Virtual Machine (JVM)

➢ The Java Application Programming Interface (Java API)

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

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries **(packages)** of related components. The following figure depicts a Java program, such as an application or applet, that's running on the Java platform. As the figure shows, the Java API and Virtual Machine insulates the Java program from hardware dependencies.



As a platform-independent environment, Java can be a bit slower than native code. However, smart compliers, wheel-tuned interpreters, and just-in-time byte Compilers can bring Java's performance close to that of native code without threatening portability.

Product features
Tomcat 3.x (initial release)

- Implements the Servlet 2.2 and JSP 1.1 specifications
- Servlet reloading
- Basic HTTP functionality Tomcat 4.x
- Implements the Servlet 2.3 and JSP 1.2 specifications
- Servlet container redesigned as Catalina
- JSP engine redesigned as Jasper
- Coyote connector
- Java Management Extensions(JMX),JSP& Strut-based administration   Tomcat 5.x
- Implements the Servlet 2.4 and JSP 2.0 specifications
- Reduced garbage collection, improved performance and scalability
- Native Windows and Unix wrappers for platform integration
- Faster JSP paring

Project Scope

Recently, a new technique will replace conventional radar systems and be deployed as part of the next generation air transportation systems. Unlike in traditional radar systems where aircraft only respond to interrogations by ground stations. In the ADS-B system, aircraft continuously obtain their positions based on some satellite positioning techniques (e.g., GPS) and periodically broadcast their positions as well as some other information such as the current velocity to ground stations and other aircraft. Recently, flight tracker Web sites based on the mashup of ADS-B data have gained popularity, providing Web users with a visual overview of air-traffic around the world.

Keywords:
APKG: Airline PKG
ASK: Airline Secret Key
PSK: Plane Secret Key
SKIDA: Airlines secret key
SKIDF: Aircraft secret key
Design and Implementation Constraints

Constraints in Analysis

- **Constraints as Informal** Text
- Constraints as Operational Restrictions
- Constraints Integrated in Existing Model Concepts
- Constraints as a Separate Concept
- Constraints Implied by the Model Structure

Constraints in Design

- Determination of the Involved Classes
- Determination of the Involved Objects
- Determination of the Involved Actions
- Determination of the Require Clauses
- Global actions and Constraint Realization

Constraints in Implementation

A hierarchical structuring of relations may result in more classes and a more complicated structure to implement. Therefore it is advisable to transform the hierarchical relation structure to a simpler structure such as a classical flat one. It is rather straightforward to transform the developed hierarchical model into a bipartite, flat model, consisting of classes on the one hand and flat relations on the other. Flat relations are preferred at the design level for reasons of simplicity and implementation ease. There is no identity or functionality associated with a flat relation. A flat relation corresponds with the relation concept of entity-relationship modeling and many object -oriented methods.

System Features

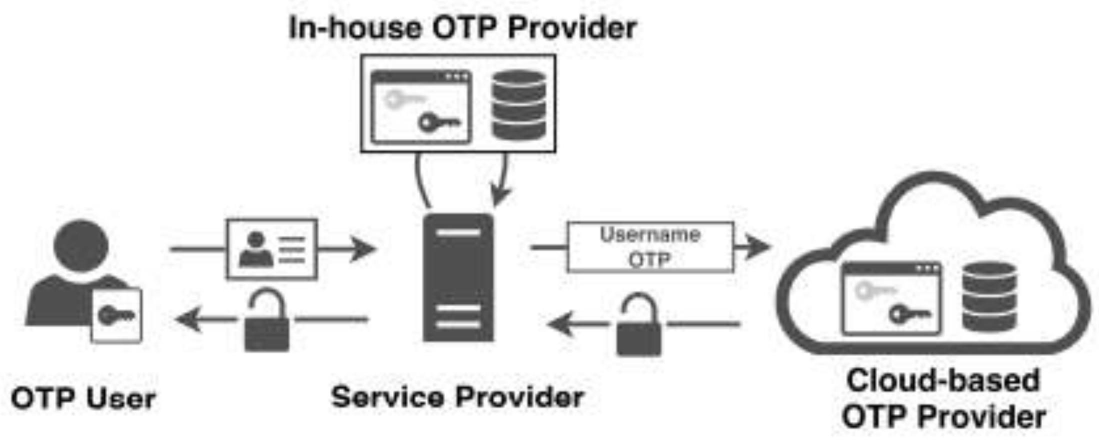
✓ Automatic Dependent Surveillance-Broadcast (ADS-B) has become a crucial part of next generation air traffic surveillance technology and will be mandatorily deployed for most of the airspaces worldwide.

✓ The number of aircraft has been increasing tremendously over the last decade. So need to verify bulk number of aircrafts using Batch verification.

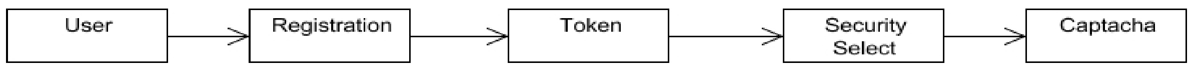
✓ We give a formal security proof for the extended scheme. Experiment results show that our schemes with batch verification are tremendously more efficient in batch verifying n signatures than verifying n signatures independently.

✓ With this accurate information, the ground controllers or other surrounding aircraft can monitor and track the location and path of an aircraft, which provides aircraft and the ground controllers a common situational awareness. This improves pilots' decision-making ability dramatically and makes air traffic management much easier.

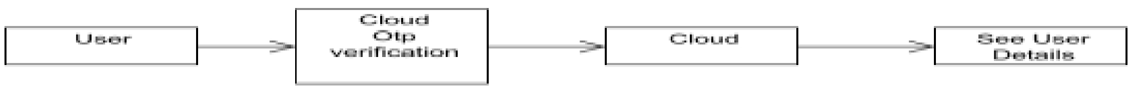
Architecture Diagram:



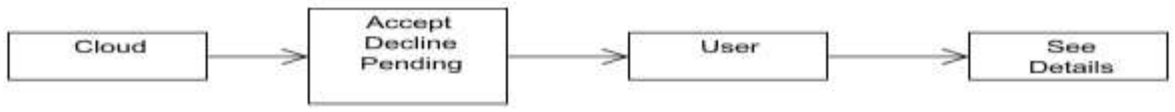
Dataflow Diagram
Level 0



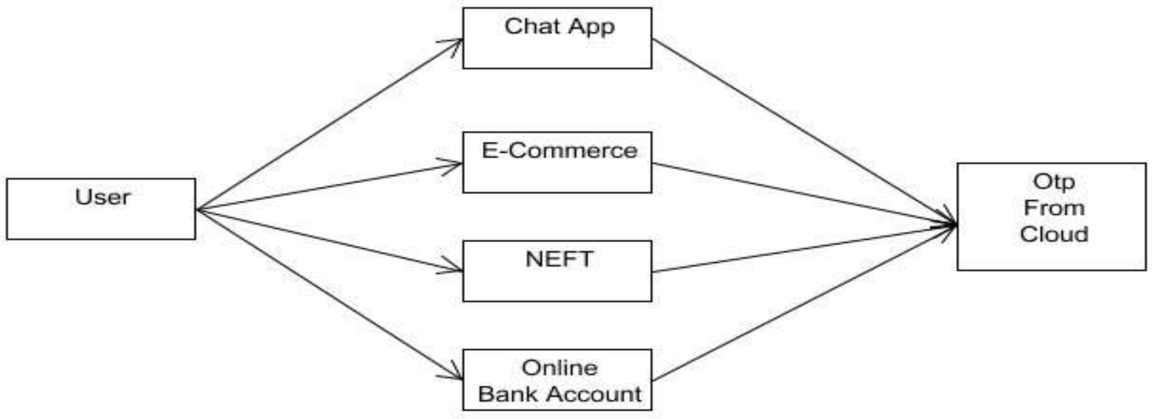
Level 1



Level 2



LEVEL3:



III. Modules

- ➢ User registration under cloud & choose the type of security provider
- ➢ Verify user details & provide the security in cloud
- ➢ Chat Application with traditional OTP as a Service and Bank account creation.
- ➢ Grocery with secure, NEFT with sensitive and Bank application with highly sensitive OTP as a Service.

1. User registration under cloud & choose the type of security provider

In this module user needs to register under the cloud by providing the basic details. After registration based on user details loud will generates one token and send that token to user email. Here after user wants to prove his genuineness by entering the valid token which was sent to user email address. If he enters wrong token his details won't take into the cloud. If user enters the valid token he will get options to choose the types of securities. After completion of security selection user wants to do captcha verification and cloud OTP verification.

2. Verify user details & provide the security in cloud

Here in this module cloud owners will get the requests from users with various types of security selections. Cloud owners will verify user details, and verify the documents step by step. If they did not find proper documents and proper details the will decline user requests. Unique users UID will be generate and send it to the user if cloud accepts the user request. Based on user security selection use can eligible to use those particular criteria applications. If he has a high security he is eligible to use all the low level security services.

3. Chat Application with traditional OTP as a Service and Bank account creation

In this module user needs to create an account in bank application and add the money into bank. A chat application that involves a traditional security it will verify the user only at registration time with the mobile number; Cloud will give one time verification to the traditional cloud OTP service. Chat application involves an option to read our text messages by taping on that particular message. And we can also send a voice message by converting voice content to text message.

4. Grocery store with secure, NEFT transaction with sensitive and Bank application with highly sensitive OTP as a Service

Grocery store is an e-commerce supermarket application with secure OTP service. User can see products and add the products into cart and removes the products from cart. User can also increase or decrease the quantity of the products in a cart. This e-commerce application avail secure and traditional features both. It will ask for opt each and every purchase. NEFT Transaction is an sensitive OTP service application which has a feature to send money to beneficiary account to instantly. And also there is a option to check the current account balance. This sensitive security avail all the features of secure and traditional OTP service. It will ask for opt each and every time of user logins to his account. Bank application contains an option called online bank account creation. For this user needs to submit kyc detail. If he already has cloud account under highly sensitive he can import all those details to bank by submitting cloud service and user UID. If he gives exact details user can get OTP. After completion of OTP all the kyc details will be imported to bank application.

IV. Conclusion

Then, a cloud-based OTP service architecture is designed. The conceptual design's security analysis shows that the architecture and the protocols are robust and sound. Possible advantages of the design are addressed both from the perspective of a service providing company and an individual service user. The architecture is a step for adoption of non-expert small to medium enterprises to safer authentication techniques than conventional ones. The proposed approach is effective as a two factor authentication security mechanism and provides many configurable options by design. User profiles are open to future development at user devices, such as regular password management, credential management, and so on. The design lets companies spend less on OTP-based TFA transition both in the perspectives of experience, employers, hardware and software. Additionally, it lets the users to manage many of their accounts easily at one place, yet via unlinkable profiles. It is believed that outsourcing OTP service in the cloud may also ease many cloud service providers bulk OTP adoption, as they do not require making additional investment. Given the aforementioned advantages, it is believed that the proposed architecture is an initial step for realization of such services.