TRANSIENT AUTHENTICATION SERVICE

Sapana V Shinde, Daniyal M Solkar, Paul Antony, Swapnil Chatte , Lakshmi Madhuri

Abstract—**Transient authentication provide a secure scheme which protects the sensitive data of the user stored on cloud. In this service a small hardware token (Mobile Phone) continuously authenticates the user's presence over a short-range, wireless link to the machine through which the user is accessing cloud. This machine in turn provides this authentication information to cloud. When the user departs, the token and machine lose contact and the machine stops exchanging the information with cloud making the data secure. We show how to leverage this authentication framework to secure sensitive and confidential data of applications.**

Index Terms—**Authentication, Bluetooth, Cloud, Service, Token, Transient Authentication.**

INTRODUCTION

Cloud solutions are scalable and ubiquitous, and follow a pay per use approach at all levels. As a concept, cloud computing's primary significance lies in allowing the end user to access computation resources through the Internet. Some scholars find cloud computing similar to grid computing [1], but some also find similarities to utilities such as water and electrical power and refer to it as utility computing [2]. Because the use of resources can be independently adjusted, it is also sometimes referred to as autonomic computing [3]. One of the main barriers to the adoption of Cloud Computing is security. User data are stored on provider servers and there is no guarantee that this information will not be accessible to a third party. This can contravene legal requirements when the stored data are sensitive, as occurs in health care or banking environments. There is no guarantee that the data will be safe and secure at the server side. Here we present a solution for secure storing of data in the cloud environment through the use of transient authentication.

I. RELATED WORK-LITERATURE SURVEY

Computing authentication requires that a user supply some proof of identity–via password, smartcard, or biometric–to a device. Unfortunately, it is infeasible to ask users to provide authentication for each request made of a device. Imagine a system that requires the user to manually compute a message authentication code for each command. The authenticity of each request can be checked, but the system becomes unusable. Instead, users authenticate infrequently to devices. User authentication is assumed to hold until it is explicitly revoked, though some systems further limit its duration to hours or days. Regardless, in this model authentication is persistent and creates tension between security and usability. To maximize security, a device must constantly re-authenticate its user. To be usable, authentication must be long-lived. Unfortunately, authentication between people and their computer devices is both infrequent and persistent. Should a device fall into the wrong hands, the imposter has the full rights of the legitimate user. Researchers at the University of Michigan have developed a new model, called "transient authentication," in which a user wears a small token, equipped with a short-range wireless link and modest computational resources. This token is able to authenticate constantly on the user's behalf. It also acts as a proximity cue to applications and services; if the token does not respond to an authentication request, the device can take steps to secure itself. This technology provides an improved method and system to maintain application data security on machines that are running or have been suspended. Applications are protected transparently by encrypting in-memory state when the user departs and decrypting this state when the user returns. This technique is effective, requiring just seconds to protect and restore an entire machine. In the second embodiment, applications utilize an API for transient authentication, protecting only sensitive state. Ports of three applications, PGP, SSH, and Mozilla are described with respect to this API.

1. WHAT IS AUTHENTICATION?

Authentication is any protocol or process that permits one entity to establish the identity of another entity. Living creatures have been performing authentication at some level for eons. The old methods of authentication are based on the realities of our physical world; basic human authentication is done by identifying unique physical characteristics of other human beings. We most commonly use facial recognition or voice recognition to identify others, but we might also use general appearance, such as style of dress or body language and actions in face-to-face situations. When nearly all human transactions are accomplished face-to-face, these methods are usually reliable, or at least, reliable enough for the purposes of most individuals. In non-face-to-face situations, people use other methods, such as basic handwriting recognition or stylistic recognition (for example, a person's writing or painting style) to authenticate a person, their possessions, or their work. To review, humans have used authentication, and throughout history, they have used three methods. These methods are:

- Something you know (the password).
- Something you have (the general's ring or seal).
- Something you are (your face, voice, fingerprints, or DNA)

2. HOW DO WE USE THESE FACTORS IN COMPUTER AUTHENTICATION

1. PASSWORDS:

Most commonly, computers use passwords, the "something you know" factor, for basic authentication. In the early days of computing, computers were large beasts, which meant they were expensive to buy and expensive to run. The computer's owner allowed others to use his computer, but he often demanded some sort of fee to offset the cost of running the system. Also, with so many people using the system, they liked keeping their own data and programs available for their own use. So, to provide proper accounting for system resource charge-back and to provide basic security for individual users' data, administrators began using usernames along with passwords, so that users could be tracked individually for authorization and accounting purposes. Passwords are the simplest authentication model to implement, and that is why password models are so common. Unfortunately, password models are also the weakest authentication model because passwords are guessed or stolen relatively easily. Users often choose weak passwords; the example of "Joe" account illustrates this. Joe doesn't want to be burdened with creating and maintaining a relatively secure password; he simply wants to log in and do his work. So, he has the username "Joe" and he creates the password "Joe" (or "joe1") for his account. This sort of weakness, being so common, makes any password model vulnerable. The system administrator can take measures to mitigate the threat of "Joe" accounts. She could implement a minimum password length, such as eight characters, for a password to be accepted by the system. She could enforce basic password complexity, which would evaluate a new password against specific criteria such as using at least one letter, one number, and one special character (!, @, #, $, %, ^, and so on). She could force password changes every so often, so that a stolen or guessed password is no longer usable after a certain period of time, as well as a password history that would prevent the same passwords from being used over and over by any individual user. However, these measures can force users to write passwords down, which limits the value of the password because it can be more easily stolen.

2. TOKENS

Some authentication systems commonly use *tokens,* which is any device or object that can authenticate a user. In the previous example, we referred to the general's ring or seal. These are examples of tokens. Common modern examples include physical keys, proximity cards, credit cards, or ATM cards. Tokens are good because they're simple. Physical keys, for example, are widely supported and cheap to produce and use. In computer authentication, cryptographic keys may be used, particularly in remote protocols such as SSH (secure shell). The advantage of cryptographic keys for remote protocols is that they may not only be used for user authentication, but also for message authentication and encryption of data in transit. Tokens have their own weaknesses, however. Because tokens are simple and cheap to produce, they are also simple and cheap to *reproduce.* This makes them vulnerable to being counterfeiting. Also, because they are typically a physical object or device, they can be stolen more easily than passwords. For this reason, tokens are typically used with another method, such as a PIN code, to reduce their usefulness if stolen.

3. MESSAGE AUTHENTICATION

It isn't always a user who must be authenticated in the computer world. Sometimes, a message must be authenticated, or at least verified that it has not been altered in transit. This is known as integrity. In this case, a message authentication code (MAC) may be used. A MAC is generated by combining the message with a secret key shared by both the sender and receiver of the message. When he creates a message for transmission, the sender generates a MAC and attaches it to the message. When the receiver receives the message, she recalculates the MAC with the same secret key, then checks her calculated MAC against the MAC attached to the message. If they match, she knows the message has not been altered. What happens if the message not only needs to be checked for tampering, but also needs to be verified because of the purported sender who actually sent the message? What if the sender later tries to claim that he never sent the message? This provides authenticity and non-repudiation, and to accomplish this, you must use digital signatures, which are a benefit of using a public-key cryptosystem such as PGP. The sender generates a hash from the message using a standard hash algorithm. He then encrypts the hash with his private encryption key and attaches the encrypted hash to the message. The receiver receives the message and encrypted hash. She then decrypts the hash with the sender's public encryption key and recalculates the hash from the received message. If her calculated hash matches the hash decrypted from the message, she can be certain that the message was not altered and that it definitely came from the sender.

II. EXISTING SYSTEM

For existing system consider the HSBC Online banking Application. The HSBC makes use of the device shown in following fig –

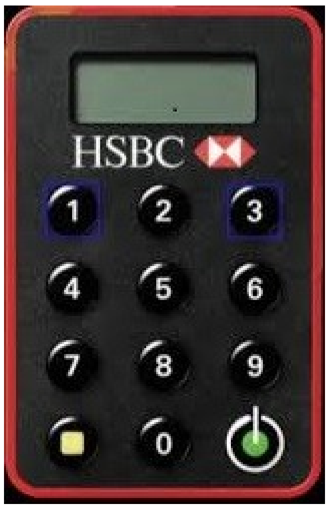

Fig1.HSBC Transient device used during authentication

The user availing the online banking facility is provided with following device. Each time user wants to login for accessing his account through online banking application he must have this device and should press the green button shown in figure. After the user presses this button he is given a code

which he should enter and only then he is granted access to application.
Some of the problems the user may face are:
1. The user to authenticate himself must each time manually enter the code.
2. If the user losses his device / the device is broken the user has no means to gain access to account until he gets a new device.
3. The user each time has to carry additional this device where ever he goes to have access to his account.

III. PROPOSED SYSTEM

The proposed system is built to overcome this problems. The proposed system will authenticate a user through a transient device .The device in this can be any device which has Bluetooth capability which the user can select of his own choice during registration. The best case is mobile device as in today's world almost each and everyone has a mobile device and carries along with him every time. Also purpose of this project is to deploy the service over cloud. Cloud computing moves the application software and databases to the large data centres, where the management of the data and services may not be fully trustworthy. This unique feature, however, raises many new security challenges which have not been well understood. In cloud computing, both data and software are fully not contained on the user's computer. Data Security concerns arises because both user data and program are residing in provider premises.

IV. ARCHITECTURE

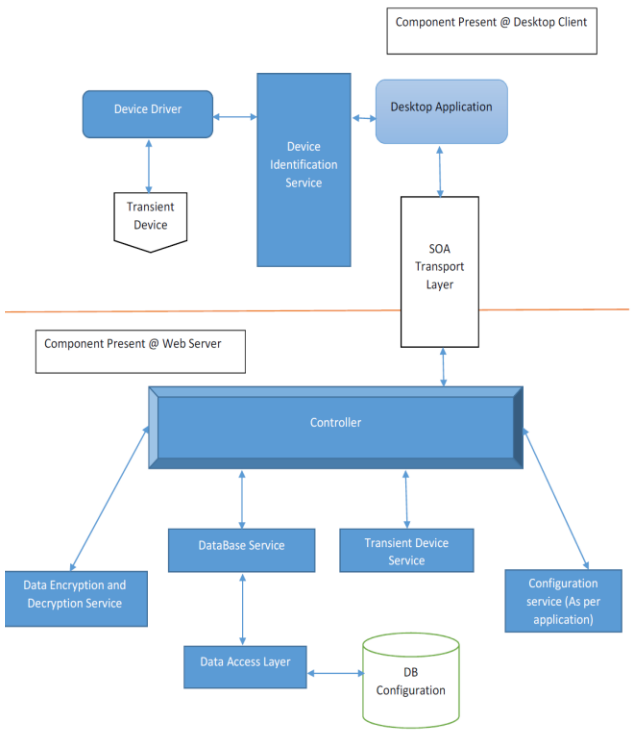
Component Present @ Desktop Client
Device Driver
Device Identification Service
Desktop Application
Transient Device
SOA Transport Layer
Component Present @ Web Server
Controller
DataBase Service
Transient Device Service
Data Encryption and Decryption Service
Configuration service (As per application)
Data Access Layer
DB Configuration

Fig2.Architecture for proposed system.

V. MODULES

1. Client

Proposed system is applicable for thin as well as thick client. For demonstration purpose we will have both the type of client which are consuming the same services. In our proposed work our focus is to develop the security services and infrastructure.

2. Transient Device Identification System

We are considering our mobile [which will have Bluetooth] as transient device , to identify the device which is connected to the system .We are using Apache Bluecove framework which is having capability to talk with the underline Bluetooth device driver and transient device identification module. The main aim of the module is to get the device identification number. This module will contain one web service, whenever consumer will call that web method it will provide the list of number of the connected Bluetooth devices. The work will be carried by consumer.

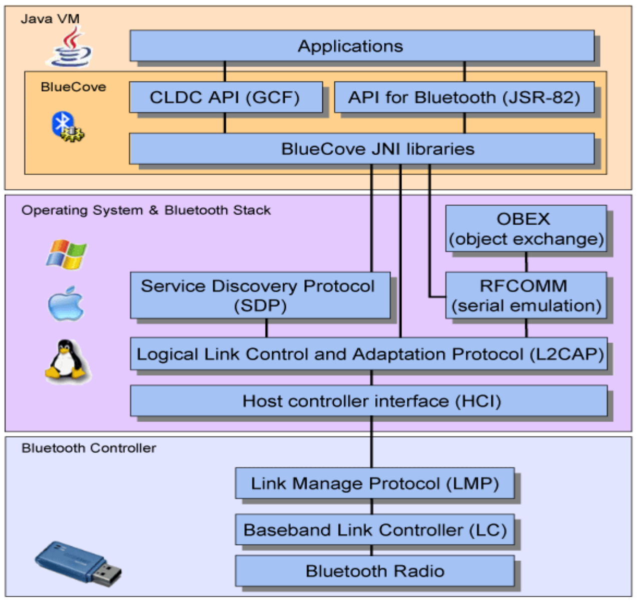
Java VM
Applications
BlueCove
CLDC API (GCF)
API for Bluetooth (JSR-82)
BlueCove JNI libraries
Operating System & Bluetooth Stack
OBEX (object exchange)
Service Discovery Protocol (SDP)
RFCOMM (serial emulation)
Logical Link Control and Adaptation Protocol (L2CAP)
Host controller interface (HCI)
Bluetooth Controller
Link Manage Protocol (LMP)
Baseband Link Controller (LC)
Bluetooth Radio

Fig3.Apache Bluecove Framework

3. Controller

Controller is the most important module in the system because it decide the work flow of the system. This part of the system will have the reasonability of communication between all the modules. This controller will be only one end point which is visible to consumer, this will also work as an abstraction layer for the web service cluster. This controller is configurable with the help of customization as required. The most important point and unique feature in the controller that without code change you can add a new work flow or modify the existing one. Controller will recognize the configuration file will behave accordingly. We are deploying the jax-ws java web service on the server machine and for customization tool we are using java jaxb [java xml data binding framework].

4. Web Services

In these module all our system business logic is present. Below are list of web services that we are having in our system.

1. Encryption and Decryption Service [configurable to any security algorithm]
2. Data Base Service [To store the data on the data base]
3. OTP Service [to generate the one time password in case of device lost case]
4. Email send service [Job is to send an email]
5. Transient device verification system
6. Login verification system.

VI. Implementation Details

The proposed system requires a client to have a Bluetooth device and a system through which he accesses the web services .So its required to have an arrangement in such a way that a system authenticates with users device and there should be communication established between both so that a system can validate the users transient device .This functionality is provided by Bluecove Java Library for Bluetooth. Bluetooth is a wireless protocol for local communications. JSR 82 is the standard Bluetooth definition for Java devices, Bluecove is an open source implementation for Windows, Mac and GNU/Linux.

While many similarities exist between Internet and Bluetooth protocols, one key difference is that Bluetooth devices move in and out of radio frequency range. Normally, before commencing communication, devices must discover others to detect the address and services that are provided by other devices.

Key programming issues

Bluetooth communications are part of the Generic Connection Framework (GCF).

Discovery-Devices must be set to discovery mode.

Known-Communication with a known remote device is established by:

Btspp -Several protocols are possible, RFCOMM is connection-oriented, and Btspp is Bluetooth Serial Port Profile.

Address - Bluetooth devices have a 12 hexadecimal digit address (e.g. 0012D25ABDE4)

Port-The port on which the remote device will receive data.

Unknown-Usually the service desired (email, chat, etc.) is known but not device address and port. Discovery can determine the address and port used by a service (i.e. the URL). More follows.

UUID

Services are uniquely identified by a Universally Unique Identifier (UUID), a 128-bit number, in hexadecimal.

Service Discovery Protocol (SDP)

SDP determines the URL given a service UUID.

Server

- advertises the UUID
- waits for a connection

Client

- Runs a discovery agent that searches for the service UUID.
- Agent reports the URLs that provide that service,
- Client connects to a URL

The encryption of stored user's credentials is important. As our data over cloud is stored on third party which we can't trust .So before storing data it must be stored in an encrypted format so that it is not readable as encryption is a process which changes the plain text in non-readable format. To do so we are using AES algorithm. The AES Algorithm is considered to be the secured algorithm for encrypting data. The overall structure of AES algorithm is shown below-

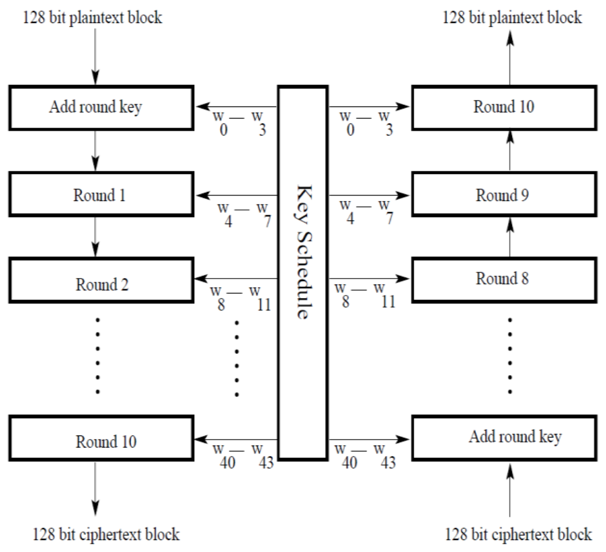

Fig4. Structure of AES Algorithm

The next figure shows the different steps that are carried out in each round except the last one.

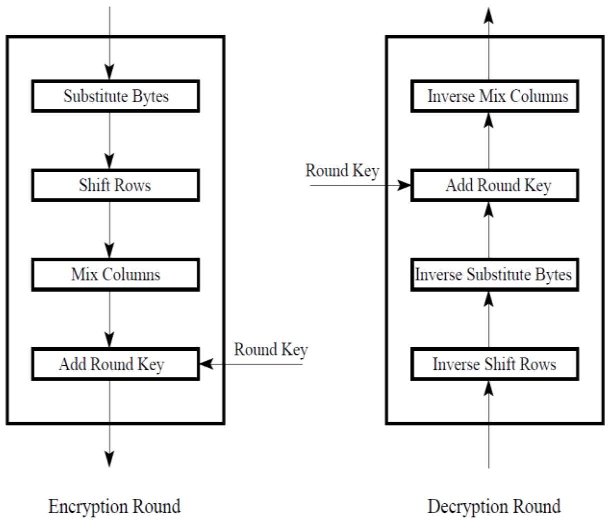

Fig5.Steps carried out in each round of AES Algorithm

1. Substitute byte- also called subbyte used for byte-by-byte substitution during the forward process.The corresponding substitution process during decryption is called InvSubBytes.

2. ShiftRows - It is used for shifting the rows of state array during forward process. The corresponding transformation during decrption process is denoted InvShiftRows for Inverse Shift Rows transformation.

3. MixColumn – Used for mixing up the bytes separately in each column during the forward process. The correspond-ing transformation during decryption is denoted InvMixColumns and stands for inverse mix column transformation.

4. AddRoundKey – Used for adding the round key to the output of the previous step during the forward process. The corresponding step during decryption is denoted InvAddRound Key for inverse add round key transformation.

The next important thing required is what happens when a user loses his transient device (mobile) or else wants to change a transient device required during authentication. To successful and securely do this, we make use of One Time password generation algorithm. The algorithm used is TOTP – time based one-time password generation.

TOTP is a variation of HOTP - Hash based one-time password generation. We define TOTP as TOTP = HOTP (K, T), where T is an integer and represents the number of time steps between the initial counter time T0 and the current UNIX time.

More specifically, T = (Current Unix time - T0) / X, where the default floor function is used in the computation.

For example, with T0 = 0 and Time Step X = 30, T = 1 if the current UNIX time is 59 seconds, and T = 2 if the current UNIX time is 60 seconds.

The implementation of this algorithm MUST support a time value T larger than a 32-bit integer when it is beyond the year 2038. The value of the system parameters X and T0 are pre-established during the provisioning process and communicated between a prover and verifier as part of the provisioning step.

VII. Conclusion

In this, we have introduced an integrated approach to achieving both diversity and multiplexing gains for system security with advanced features of transient device. The proposed approach is based on our work details a new type of authentication, called Transient Authentication, which differs from previous authentication systems.

We describe the four principles underlying Transient Authentication: tying capabilities to users, securing just faster than attackers, doing no harm, and ensuring explicit consent. We argue that these four principles are essential for providing an effective, yet usable, authentication system to ensure physical security in mobile devices. In our system, we are using cell phone as a token which will authenticate the system and the client machine, which can be a laptop or a desktop computer. These two systems are connected to each other via a Bluetooth. Once, the devices are authenticated and connected then our service will allow user to gain access to the application. When the user along with his cell phone is in the range of the laptop or desktop computer, the application will be available for access and as soon as the user is outside the range then the user is automatically logged off.

VIII. Future Enhancement

Systems such as laptops or desktops are owned by a particular user. Furthermore, the user owns very few of these devices and notices if one of these devices is lost or stolen or the username or password is well known to the unauthorized users. However, in a ubiquitous computing environment the user may be interacting with many shared devices. The user no longer knows when one of these devices has been stolen, allowing the token to provide capabilities to compromised devices. Instead of a simple binding process, requiring the user to remember which devices have been authorized, a simple user interface can inform her which devices are currently trusted. Creating user interfaces to inform the user of consent given to devices, and also devices that are currently using that consent, is a topic of future research

Several future research topics could mitigate this problem. The first is to require biometric feedback to keep the token functioning. Although a fingerprint may work for an initial authentication, continuous monitoring of heartbeat or body temperature could detect if the user removes the token. If the pulse or heat source is lost, the token must be revived with a password. Another approach is to formulate the face image or embedding the image of any user identity on the transient device without wireless connection which will detect the identification to authenticate the systems and its applications. Another possibility is to physically implant the user with a small device containing a base secret which would be more secure and complex. However, inductive wireless communication could transmit that key to a worn device, which relays the capabilities to the mobile device. There are obvious social constraints involved in implantable keys, but in certain high security scenarios it may be acceptable. If an attacker is able to insert a repeater between the user and his device, it can be held in an authenticated state. Although theoretical defenses have been proposed to defeat 92 such wormhole attacks, no one has demonstrated a practical implementation of such a defense. This may conclude the future scope for the extension to authenticate the system and its application with the advantage of facilitating the transient device.

IX. REFERENCES

[1] M. Baker, R. Buyya, and D. Laforenza, “Grids and grid technologies for wide-area distributed computing,” International Journal of Software: Practice and Experience, vol.32, pp. 1437-1466, 2002.

[2] C. S. Yeo, S. Venugopal, X. Chu, and R. Buyya, "Autonomic metered pricing for a utility computing service", Future Generation Computer Systems, vol. 26, issue 8, pp. 1368-1380, October 2010.

[3] R. Sterritt, “Autonomic computing,” Innovations in Systems and Software Engineering, vol. 1, no. 1, Springer, pp. 79-88. 2005.

[4] William Stallings, lawrieBrown,”Computer Security”, Pearson.