University of Nairobi

School of Computing and Informatics

Research Prototype

Topic
Multi-Factor Authentication and Secure Integration of Mobile Payments

Presenter
Kennedy Amwoyo (P58/70483/2008)

Supervisor
Dr. Elisha Abade

April 2014

Submitted in partial fulfilment of the requirement of a Masters of Science degree in Computer Science of the University of Nairobi
Declaration

I, Kennedy M. Amwoyo, hereby declare that this research project and the work presented in it, is my original work and has not been presented for any other University award.

Signature: ___________________ Date: _________________

Name: Kennedy M. Amwoyo Reg. No.: P58/70483/2008

This project has been submitted in partial fulfilment of the requirement of the Master of Science Degree in Computer Science of the University of Nairobi with the approval of the University Supervisor.

Signature: ___________________ Date: _________________

Name: Dr. Elisha Abade

School of Computing and Informatics

University of Nairobi
Dedication

To

My Family,
My Supervisor,
My Lecturers &
My Colleagues

For your valuable support throughout the whole process

May the Almighty God bless the works of your Hands and Minds
Acknowledgment

First and foremost, I would like to acknowledge the Lord Almighty for the Strength, Knowledge and Wisdom. Secondly, I appreciate the Support accorded by my project supervisor, Dr. Elisha Abade and the entire panellists for their Corrections and Insights. Finally, I recognise moral support accorded by my beloved wife, Elsie Wanji.
Abstract

Adoption of technology reduces cost of doing business, creating competitive advantage and efficiency. The cost of Internet connectivity and smart devices is gradually dropping, exposing many to vast online resources and business opportunities.

The above-mentioned trend is giving rise to the need to operate virtual shops so as to reach the vast online community which is not restricted to boundaries and time. This is greatly facilitated by business models such as Software as a Service as Cloud Computing continues to gain popularity day by day.

This move compels system developers address security concerns within and without the systems. System security is indeed a wide topic but we are particular interested in secure authentication schemes.

In addition to the above, there is need to integrate locally accepted payment solutions for near-realtime automatic receipting. Every business requires prompt payment for goods and services offered. Providing localised solution not only creates trust for the service but ensures quick adoption and smooth transition of the services.
Table of Contents

Declaration........................................................................................................................................... i
Dedication............................................................................................................................................... ii
Acknowledgment ................................................................................................................................. iii
Abstract ................................................................................................................................................ iv
Chapter 1: Introduction .......................................................................................................................... 1
  1.1 Background ..................................................................................................................................... 1
  1.2 Problem Statement ......................................................................................................................... 2
  1.3 Objectives ...................................................................................................................................... 3
  1.4 Significance ..................................................................................................................................... 4
  1.5 Project Justification ....................................................................................................................... 5
Chapter 2: Literature Review .................................................................................................................. 6
  2.1 Introduction .................................................................................................................................... 6
  2.2 Multi-Factor Authentication ........................................................................................................... 6
    2.2.1 Knowledge factors: "something only the user knows" ........................................................... 8
    2.2.2 Possession factors: "something only the user has" ................................................................. 9
    2.2.3 Inherence factors: "something only the user is" .................................................................... 22
    2.2.4 Cost effectiveness .................................................................................................................... 23
    2.2.5 Market acceptance ................................................................................................................... 24
    2.2.6 Man-in-the-browser ................................................................................................................. 24
    2.2.7 Man-in-the-middle attack ....................................................................................................... 25
    2.2.8 One Time Password .................................................................................................................. 25
  2.3 M-Pesa Integration .......................................................................................................................... 27
    2.3.1 Traditional E-Commerce ......................................................................................................... 27
    2.3.2 Mobile payment ...................................................................................................................... 27
    2.3.3 M-Pesa Introduction .............................................................................................................. 35
    2.3.4 M-Pesa Concept ..................................................................................................................... 35
    2.3.5 M-Pesa Services ....................................................................................................................... 35
    2.3.6 M-Pesa Integration Challenges ............................................................................................. 36
    2.3.7 M-Pesa Integration Justification ............................................................................................ 36
  2.4 Summary ....................................................................................................................................... 37
Chapter 3: Methodology ........................................................................................................................... 38
  3.1 Introduction .................................................................................................................................... 38
  3.2 Choice of Model ............................................................................................................................... 38
  3.3 Waterfall Model ............................................................................................................................. 38
Chapter 1: Introduction

1.1 Background

Every day, many Small and Medium Sized Enterprises (SMEs) and Corporate are turning to Information and Communication Technology (ICT) solutions to cut down operational costs and gain competitive advantage among others. Secure Authentication and Online Transactions is a vision of most businesses. However, the gap between the reality and the ideal is fairly wide.

For a long time, a great percentage of the local customers did not fully embrace nor trust transacting online using credit cards. Very few Kenyan companies have wallet accounts for famous online check-out systems like Paypal. With the introduction of inter-continental fiber connectivity like TEAMS and SEACOM, Internet Service Providers like Safaricom, MTN Business among others are already providing Cloud Computing Software as a Service (SaaS) business plans for their customers. As a result, SaaS business model is expected to rise exponentially.

Local software engineers are developing solutions and requesting for Application Programming Interfaces (APIs) to enable the solutions transact securely with Mobile Money Transfer Systems and Commercial Banks so as to reduce lead times. However, most of the local Mobile Money Transfer Systems do not have standardized and documented APIs for integration. This is hampering the growth of local online businesses.

With the rise of SaaS business model, there is a need to address secure logins. Authentications schemes need to be aligned with the current devices that are accessing the online portals (like tablets and smart phones), hence a paradigm shift from traditional two-factor authentications. More so, there is a need to transact securely using emerging and widely accepted innovations like M-Pesa Mobile Money Transfer System.
1.2 Problem Statement
Credit card penetration and adoption is still a preserve of developed countries and a few elite local customers. This could be attributed to Luddite syndrome, lack of information and distrust of such technologies amongst the locals. More so, despite having local online wallet solutions that integrates with Mobile Money Transfer Systems, online transactions still seems a challenge. One would attribute it to the fact that users must register and maintain online wallet accounts, a process many consider cumbersome. Third-party integrators too are charging high commissions per transaction and this increases the overall cost per transaction.

More often than not, static or single password (one-factor) authentication mechanisms are susceptible to brute-force attacks resulting to unauthorized access, given enough attempts and time. Authentication based on one factor does not provide adequate security because the static password does not change between subsequent logons or is rarely altered. This risk can be greatly reduced by continually altering the password, as offered by the One-Time Password (OTP) two-factor authentication mechanism.

Traditional two-factor authentication like RSA dongles, biometrics (iris scan and finger print readers) are being challenged by cloud computing technology which is increasingly being accessed by smart devices (like smart phones and tables). Physical sizes of mobile devices limit the number of peripherals and ports which could be attached to them. In addition to the above constraint, mobile devices (due to mobility) often experience fluctuating, limited internet connectivity hence not suitable for bandwidth-intensive authentication schemes.

Reports, according to Communications Commission of Kenya (CCK), indicate that 75% of Kenyan population has access to mobile phones. It is therefore necessary to provide phone-based multi-factor authentication so as to keep up with technological trends like Cloud Computing Software as a Service (SaaS) business models.
1.3 Objectives

The study seeks to achieve the following key objectives:

1. To Securely Integrate Mobile Money Transfer System with an online solution
2. To Secure logins using time and session constrained Random Password Multi-Factor Authentication
1.4 Significance

The outcome of the Project seeks to provide local developers and application integrators with a Prototype and sample Application Programming Interface (API) for integrating local online solutions with Mobile Money Transfer Systems like Safaricom M-Pesa. The Project also seeks to address the limitations of Mobile One Time Password 2-Factor Authentication using Random Password that is browser Session and Time constrained.
1.5 Project Justification

- **Distributed Systems**
  The ability to integrate various systems, running on different platforms using standard communication protocols (HTTPS, SMPP, SMTPé) and message parsing mechanisms

- **Information System Security**
  The ability to illustrate the use of session tokens, one-way string encryption and generation of random (session and time constrained) passwords for multi-factor authentication

- **E-Commerce**
  The ability to integrate with Mobile Money Transfer System as the next generation local e-commerce solution

- **Mobile Networks**
  The ability to integrate with Short Message Peer-to-Peer (SMPP) gateways for sending Short Message Service (SMS) notifications and tracking message deliveries
Chapter 2: Literature Review

2.1 Introduction

Manual operations are time consuming hence reducing the efficiency and effectiveness of an organization, making it less competitive. As a result, such organizations are adopting computer-based Information Systems to address those challenges. Such Information Systems have increased with the new innovations in Information Communication Technology (ICT) hence improving quality of service and productivity with limited resources.

However online Information Systems are open to the public making them vulnerable. Some of the risks include: cyber crime, espionage, sabotage, identity theft among others. With the risks involved, there is need to address security issues especially on user authentication.

Most Online Systems use Static Password or Single (One-Factor) Authentication, which is susceptible to brute force attack. One-Factor Authentication does not provide adequate security since the static password remains the same with subsequent logons unless a user manually changes it.

2.2 Multi-Factor Authentication

Multi-factor authentication (also MFA, two-factor authentication, TFA, T-FA or 2FA) is an approach to authentication which requires the presentation of two or more of the three authentication factors: a knowledge factor ("something only the user knows"), a possession factor ("something only the user has"), and an inherence factor ("something only the user is"). After presentation, each factor must be validated by the other party for authentication to occur.

Two-factor authentication is commonly found in electronic computer authentication, where basic authentication is the process of a requesting entity presenting some evidence of its identity to a second entity. Two-factor authentication seeks to decrease the probability that the requester is presenting false evidence of its identity. The number of factors is important, as it implies a higher probability that the bearer of the identity evidence indeed holds that identity in another realm (e.g., computer system vs real life). In reality, there are more variables to consider when establishing the relative assurance of truthfulness in an identity assertion than simply how many "factors" are used.
Two-factor authentication is often confused with other forms of authentication. Two-factor authentication requires the use of two of the three authentication factors. The factors are identified in the standards and regulations for access to U.S. Federal Government systems. These factors are:

- Something only the user knows (e.g., password, PIN, pattern);
- Something only the user has (e.g., ATM card, smart card, mobile phone); and
- Something only the user is (e.g., biometric characteristic, such as a fingerprint).

Two-factor authentication is not a new concept, having been used throughout history. When a bank customer visits a local automated teller machine (ATM), one authentication factor is the physical ATM card the customer slides into the machine ("something the user has"). The second factor is the PIN the customer enters through the keypad ("something the user knows"). Without the corroborating verification of both of these factors, authentication does not succeed. This scenario illustrates the basic concept of most two-factor authentication systems: the combination of a knowledge factor and a possession factor.

Two-factor authentication (or multi-factor authentication) is sometimes confused with "strong authentication", however, "strong authentication" and "multi-factor authentication" are fundamentally different processes. Soliciting multiple answers to challenge questions may be considered strong authentication but, unless the process also retrieves "something the user has" or "something the user is", it would not be considered two-factor authentication. The U.S. Federal Financial Institutions Examination Council issued supplemental guidance on this subject in August 2006, in which they clarified, "By definition true multifactor authentication requires the use of solutions from two or more of the three categories of factors. Using multiple solutions from the same category ... would not constitute multifactor authentication."

Other factors under debate for inclusion are time and location. For example, two users (even perfect twins with cloned knowledge and tokens) cannot be in the same place at the same time.
2.2.1 Knowledge factors: "something only the user knows"

Knowledge factors are the most common form of authentication used. In this form, the user is required to prove knowledge of a secret in order to authenticate.

- **Password**
  A password is a secret word or string of characters that is used for user authentication. This is the most commonly used mechanism of authentication. Many two factor authentication techniques rely on password as one factor of authentication.

- **PIN**
  A personal identification number (PIN) is a secret numeric password and is typically used in ATMs. Credit and ATM cards do not contain the PIN or CVV on the magnetic stripe. This aligns with the principle that the PIN is not part of "something the user has" for this use.

- **Pattern**
  Pattern is a sequence of cells in an array that is used for authenticating the users. e.g. Pattern based authentication is used in Android devices at login.
2.2.2 Possession factors: "something only the user has"
Possession factors have been used for authentication for centuries, in the form of a key to a lock. The basic principle is that the key embodies a secret which is shared between the lock and the key, and the same principle underlies possession-factor authentication in computer systems.

There are several ways of attacking such a system, including:

1. An attacker can determine the shared secret, for example by attacking the authenticator or a management system, reverse-engineering the possession factor, or intercepting the secret during authentication. In the case of a lock and key, the lock can be picked. In an inadequately secured computer system, for example, a database containing the shared secrets can be attacked through SQL injection.

2. An attacker can steal the possession factor. In the case of a lock and key, the attacker can steal the key and use it before the rightful owner notices the loss and has the lock changed. In the case of a computer system, the attacker can steal the possession and use it before the rightful owner notices and has the device cancelled.

3. An attacker can copy the possession factor while it is inadequately safeguarded. An attacker can take an impression of a physical key and make a duplicate; and in the case of computer systems, can clone the possession factor.

4. The attacker can intercept the authentication process and masquerade as the authenticator to the party seeking authentication and vice versa, in a man-in-the-middle attack. In the case of the lock and key, the attacker can interpose a dummy lock that will allow them to make a copy of the key then later use the copy in the real lock. In the case of a computer system, the attacker can for example interpose a counterfeit authentication interface to intercept the communications and relay the authentication information between the legitimate user and the real authenticator.

5. The attacker can hijack access after authentication. In the case of a lock and key, the attacker can wait until the owner of a key has opened the lock, and then gain access to the locked facility. In the case of a computer system, the attacker can for example use man-in-the-browser malware, session fixation, or side jacking to gain access to a secured facility as soon as a legitimate user has logged in.

The security of the system therefore relies on the integrity of the authenticator and physical protection of the possession factor. Copy protection of the possession factor is a bonus. This may comprise some form of physical tamper resistance or tamper-proofing, it may use a challenge/response to prove knowledge of the shared secret whilst avoiding risk of
disclosure, and it may involve the use of a pin or password associated with the device itself, independent of any password that might have been demanded as a first factor. A challenge/response will not defeat a man-in-the-middle attack on the current authentication session but will prevent the attacker from successfully reusing or replaying credentials.

The secret may simply be a number, large enough to make guessing infeasible or it may be a secret key embodied in an X.509 certificate, supported by a PKI.

Many commercial and a few non-commercial solutions are available for providing the possession factor as described in the following sections. The system designer must consider various trade-offs, such as between costs of deployment and support, usability and user acceptance, and hardware and software requirements. Physical tokens may authenticate themselves by electronic means (e.g. a USB port) or may display a number on a screen, derived from the shared secret and which the user has to type in. In the former case, device drivers may be required which the system designer may or may not be able to rely on if he has no control over the client device (as in the case of authentication to a public website). A one-time pad (such as PPP, described later) is a little different but can still be classed as a possession factor.

**Tokens with a display (disconnected tokens)**

*Fig. 1 - RSA SecurID token*

**RSA SecurID token**

A number of types of pocket-sized authentication token are available which display a changing passcode on an LCD or e-ink display, which must be typed in at an authentication screen, thus avoiding the need for an electronic connection. The number is derived from the shared secret by a cryptographic process which makes it infeasible to work out the secret from the sequence of numbers. Essentially, the secret is hashed or otherwise
cryptographically combined with a challenge, and the result is displayed. The same process repeated on the authentication server will yield the same result if the correct secret was used. The challenge can take one of three forms:

- In a “sequence-based” token, the token may have a button that is pressed to switch it on and display a new pass code. The cumulative number of button pushes can be used as the challenge. The server, however, must assume that the button may have been pressed a number of times since the last actual use, and attempt the authentication with all likely numbers of button pushes.

- In a “time-based” token, the token generally contains a quartz time source, allowing the absolute time to be used as the challenge and a new pass code to be displayed (usually) every 30 or 60 seconds. In this case, the authentication server must allow for a drift in the time source by trying the authentication with a previous and subsequent time as well as the current time. It can hence keep track of the drift in the clock.

- The token may have a small keypad on which a challenge can be entered. This may either be a fixed PIN assigned to the user, or a challenge generated by the server and displayed at the authentication screen, or both.

Most such tokens have at least a basic level of copy protection in that it would take a certain level, perhaps a high level, of sophistication to extract the secret from the chip on which it is stored.

Display tokens have the advantage that no drivers or electronic interfaces are required on the user access device. Often, it is possible to arrange for the pass code from the display to be appended to a password in an existing password field, so that the only modifications required are in the authentication server. A disadvantage in some sectors is that the display is usually small, and may be difficult to read for visually impaired users.

There are several manufacturers of display tokens used to authenticate online transactions. These are generally designed as a keyfob to be attached to a key ring or lanyard, or as a device that can conveniently be carried in a pocket or handbag.

Recently, it has become possible to take the electronic components associated with regular keyfob tokens and embed them in a credit card form factor. However, because card thickness (0.79 mm to 0.84 mm) prevents traditional components or batteries from being employed, special polymer-based batteries must be used which have a much lower battery life than their traditional coin cell brothers, and an e-ink rather than LCD display is generally used.
Additionally, low-power semiconductor components are necessary to conserve the power used during sleep and/or actual use of the product.

Disconnected tokens are vulnerable to man-in-the-middle attacks by virtue of the fact that they are physically disconnected from the authenticating entity.

**Connected tokens**

Like display tokens, connected tokens embody a shared secret (either a long number or in some cases an X.509 certificate). This is normally interrogated by a challenge/response to avoid exposing it. Most types of connected tokens have at least some level of copy protection. For electronic tokens that require a physical connection (as opposed to wireless), the form-factor of the electronic interface used to connect the token to the machine (card-reader for magstripes and smartcards, USB port, audio port) sets a minimum size for the token. Because the necessary electronics are miniaturised, end-user resistance to carrying around authentication-tokens based on their size, bulk, or weight is uncommon (ATM cards and driver's licenses are the most common examples). Some types of tokens require a device-reader, possibly with special device drivers, whereas others use an interface which is almost universally available such as USB. Even USB-based tokens, however, may not be available on highly locked-down terminals such as thin clients, pads, or kiosk systems.

**Background:** In a business or government environment, where the details of the user-access-device (connected-token hardware) and its capabilities are known, and any required device drivers are deployed in advance, an authentication-token may be more convenient than a pin or a passphrase, as there is no need to type a passcode. Note that when such a token replaces a password, one type of single-factor authentication (knowledge-based) has been replaced with another (possession-based). This is distinct from two-factor authentication, where more than one mode of authentication is required, such as requiring both an authentication-token, plus a password or PIN. Adding token-based authentication to a password-based authentication system monotonically improves overall security, but reduces overall convenience, and increases overall maintenance and replacement costs. Replacing passwords with tokens can improve overall security, but can also reduce overall security, depending on the defenses against specific threats relevant to the authentication scheme in question (passwords can be compromised by keyloggers, the adversary seeing a written version, the user telling their password to someone else ... passwords can be denied if the user forgets the password or if the user/adversary enters too many wrong passwords inducing lockout and/or
ruins the keys used for password-entry ... tokens can be compromised by being stolen/lost, by being cloned, or by the user giving their token to someone else ... tokens can be denied if the user loses them or the user/adversary ruins the token and/or token-reader). Replacement of a compromised or lost physical token is considerably less convenient (hardware and software) than replacement of a compromised or forgotten password (software only).

Because passwords are supported by multiuser operating systems which finally became ubiquitous between by circa 2005, and because connected-tokens are used commercially as a convenient single-factor possession-based replacement for single-factor knowledge-based passwords -- although ironically replacement of a lost token is significantly more inconvenient than resetting of a lost password -- many two-factor authentication systems rely on passwords as one factor, and connected tokens as the other factor. This is a classic model used by ATM machines, which rely on possession of a magstripe card, plus knowledge of a PIN for large transactions (or at some petrol pumps a zipcode). In a certain sense, most business and governmental facilities have always used a kind of two-factor authentication system. In order to login to the computer (since the 1970s for mainframe-class hardware or since the 1990s for microcomputers as used in enterprises), one must have the password... but in order to access the computer in the first place, one must also have a magstripe card (or a door key) to gain entry to the building and/or the office. Traditionally, this type of only-authorized-personnel-get-in-the-door followed later by enter-your-personal-password-to-login security is not treated as true two-factor authentication, because in the usual case, more than one person is permitted by the first type authentication (any employee can get in the door), whereas the second type of authentication is specific to a particular person (only the individual employee knows their personal password). In small businesses, the line becomes even fuzzier: getting in the door is usually governed by whether the secretary lets you pass (rather than by possession of a magstripe card), and computers may use a single company-wide password (or have the password written on a sticky-note in plain sight).

This article deals with the traditional academic subject of multi-factor authentication, where access to a particular device is governed by logically simultaneous multi-factor authentication to the device by a single person (rather than logically distinct actions that are significantly separated both temporally and spatially). In plain terms, if in order to login to a computer the employee must swipe an employee-specific magstripe card through a card reader affixed to the keyboard, and then enter their passphrase via that keyboard, then the computer in question
requires multi-factor authentication (two factor in this example). If instead, the employee swipes a company-wide magstripe to get into the door of the building, goes up the unsecured elevator, goes through the unlocked door to their office, and enters their employee-specific password to login, then the computer is traditionally said not to have multi-factor authentication, in terms of analyzing the computer's security. Analyzing overall security is a fuzzier question; in particular, if the employee has a private office, with a locked door to which only they have the key, and inside their computer requires a password only they know, a security analysis of the computer's overall defenses might consider it to be a two-factor security system, or might not, depending on the analyst.

**Magnetic stripe cards**

Magnetic stripe cards (credit cards, debit cards, ATM cards, loyalty cards, gift cards, etc.) are easily cloned and so are being or have been replaced in various regions by smart cards, particularly in banking. However, even though the data on the magnetic stripe is easily copied, researchers at Washington University in St. Louis have found that the random and unique disposition of the billions of individual magnetic particles on each magnetic stripe can be used to derive a "magnetic fingerprint" which is difficult, but not impossible to clone. This is an example of a physically unclonable function. Special magnetic card readers have been developed and commercialised under the name "Magneprint", which can digitise this fingerprint in order to positively identify an individual card.

An advantage of this system is that a magnetic fingerprint already exists on every magnetic stripe card, being an intrinsic characteristic, and so no cards would need to be re-issued in order to upgrade an existing system. Each swipe of the card provides a correlative number called a dynamic digital identifier that can be scored and "matched" to the originating value to determine the cards authenticity. Since the number changes each time, it cannot be re-used as long as all processing is authenticated. It does require a special reader that can read the magnetic fingerprint value, but these readers can be swapped out incrementally as old readers wear down. So the actual investment could be incorporated as an incremental increase (due to licensing, increased equipment complexity, etc.) of current business cost expectations.

**Smartcards**

Smart cards are the same size as a credit card. Some vendors offer smart cards that perform both the function of a proximity card physical access device and network authentication.
Users can authenticate into the building via proximity detection and then insert the card into their PC to produce network logon credentials. In fact, they can be multi-purposed to hold several sets of credentials, as well as electronic purse functionality, for example for use in a staff canteen. They can also serve as ID badges.

In some countries, notably in Europe and Asia, banks and financial institutions have implemented Chip Authentication Program technology which pairs a banking smart card with an independent, unconnected card reader. Using the card, reader and ATM PIN as factors, a one-time password is generated that can then be used in place of passwords. The technology offers some support against transaction alteration by facilitating Transaction Data Signing, where information from the transaction is included in the calculation of the one-time password, but it does not prevent man-in-the-middle attacks or man-in-the-browser (MitB) attacks because a fraudster who is in control of the user's internet or is redirecting the user to the legitimate website via a hostile proxy may alter the transaction data "in-line" before it arrives at the web-server for processing, resulting in an otherwise valid transaction signature being generated for fraudulent data.

As has already been indicated, there are two kinds of smart card: contact smart cards with a pattern of gold plated contacts, and contactless or proximity cards, with an RFID chip embedded within the plastic. The former are more often used in banking and as a 2nd factor, and can be conveniently carried with other credit/debit/loyalty cards in a wallet. They are normally loaded with an X.509 certificate. However, they do need a special reader. Some laptops and thin client terminals have a smart card reader built in, and PCCard smart card readers are available which can be kept permanently within the shell of the laptop. Alternatively, USB smart card readers are available which are no more expensive than many display tokens, in fact, some smartcards have an interface which is electrically (but not mechanically) USB, so that the reader needs no intelligence whatsoever and consequently can be very cheap. Even so, it is less convenient than a built-in or PCCard reader, but is a good option for a desktop computer.

MS Windows has smart card authentication functionality built in, allowing authentication against a password and a smart card with no additional software apart from the smart card device driver (if needed). This can be configured to screen-lock the computer if the smart card is withdrawn. If the card also has a contactless chip used for physical access control, the
user will be forced to lock his screen by withdrawing his smart card each time he leaves the office.

There have also been smart cards released over the last five years which employ a combination of an embedded 2FA token inside a credit card form factor. These "powered" smart cards typically consist of:

- An ISO compliant credit card (ID-1) size
- A flexible display
- A switch-on button.
- An embedded non rechargeable battery.

When the button is pressed, the card displays an OTP value, which is then typed by the user on his PC keyboard. On the remote application side, the OTP number is checked using the authentication server. The OTP number is calculated according to the OATH industry standard and using some secret data securely stored in the device.

Another concern when deploying smart cards, USB tokens, or other TFA systems is the security of the software loaded on to users' computers. A token may store a user's credentials securely, but the potential for breaking the system is then shifted to the software interface between the hardware token and the OS, potentially rendering the added security of the TFA system useless.

The downsides of smart cards include that they are not the smallest form factor (although they do fit conveniently in a wallet) and that the card reader is an extra expense. Another disadvantage is that they are less robust than most other forms of token. Repeated flexing can damage both contact and contactless smart cards, and adverse climatic conditions can reduce the reliability of contact smartcards.

**Wireless**

RFID-based tokens exist. Bluetooth-based tokens exist. Contactless smart cards (a wireless version of the traditional smartcard) exist.

**USB tokens**

A USB port is standard equipment on today's computers, and USB tokens generally have a large storage capacity for logon credentials, and perhaps user data as well. However, they may be relatively costly to deploy and support, are vulnerable to theft and fraud, and have
met user resistance. Any USB memory device can be used as a token simply by storing a secret (possibly an X.509 certificate) on it, but then there is nothing to stop it from being copied. This can be prevented if the device is designed to present itself as an authentication device responding to a challenge/response protocol rather than as a storage device. The downside is that a special device driver may then be required.

**Audio Port tokens**

Audio port tokens are usually used to provide authentication service for mobile terminals, because many different mobile manufacturers have various own interface, such as idock, micro USB, mini USB and etc. In contrast the audio port is the most standard port on today's smart mobile terminals, and the audio port can be used for data transfer between authentication tokens and mobile terminals instead of USB port. An audio port token usually has a built-in battery as a power supply for the tokens. It has almost same function as the USB tokens except mobile terminal support, which is a digital certificate container with onboard encrypt/decrypt and sign/verify function.

**Soft tokens (computer-simulated software-based tokens)**

The functionality of any disconnected token can be emulated as a "soft token" on a PC or smartphone using deployed software, whereupon that device itself becomes the possession factor. This saves on deployment costs, but against that, the secret is vulnerable to any attacker or malware that can gain full access to the device. The Zeus Trojan, which can now infect mobile devices running Android or BlackBerry OS, specifically targets banking credentials and may forward them to the attacker at a website set up for the purpose, or by SMS messaging.

The secret may comprise an SSL client certificate which can be used to authenticate the device (PC or smartphone) on which it is stored, and may be used directly to authenticate the client in an SSL connection. Whilst stored on the device, even if held in a password protected certificate store, it is still potentially vulnerable to theft by malware as the certificate store has to be unlocked to be used. Indeed, the malware might trick the user into revealing the password or steal it by keystroke logging.

Such client certificates can be stored more securely in the Trusted Platform Module (TPM) chip, fitted to many modern laptops. This is tamper-resistant and requires a password or
passphrase to unlock it, and contains a cryptographic processor capable of challenge/response processing without divulging the secret.

**One Time Pad**
A one-time pad is a password used only once. Schemes based on a one time pad have been described but are rarely deployed due to the need to supply a new password (or 'pad') for each authentication.

Schemes which use a grid-card are not one-time pads, and are akin to requesting a selection of characters from a password *known* by the user (albeit a password written down). As such, they only protect against re-play attacks (as the same selection of characters can't be sent) and not against duplication of the entire grid (or the building up of a duplicate answer grid over time).

**Mobile Phone**
There is presently only limited discussion on using wired phones for authentication; most applications focus on use of mobile phones instead. A new category of TFA tools transforms the PC user's mobile phone into a token device using SMS messaging, an interactive telephone call, or via downloadable application to a smartphone. Since the user now communicates over two channels, the mobile phone becomes a two-factor, two-channel authentication mechanism. Newer solutions making use of secret photographs to block phishing introduce a third layer of security, two-way (e.g.: mutual) authentication, or later still is the ability to use a QR_code scanned by a smartphone as your two-factor authentication, such as the HOTPin system. Recent examples include Google's two-step verification option, Toopher and CryptoPhoto.

**Vulnerability to attacking**
Any authentication process which utilizes an out-of-band method is inherently vulnerable to classic man-in-the-middle (MITM) attacks. Note: contrary to some assertions, mutual authentication becomes almost useless in an out-of-band scenario since the man "in the middle" fraudster can simple forward the mutual authentication indicators to the victim with the rest of the information being communicated. Mutual authentication only works in a connected environment, in which the identity information being transmitted back to the user
can, itself, be validated by the recipient or, in the alternative, re-transmitted back to the authenticating entity where it is verified.

MITM terminology can apply to scenarios where the adversary is merely an eavesdropper on the data being transferred (sending a password over unencrypted IM ... or sending a credit card through the mail with the PIN number in the same envelope ... are modern examples of this scenario, but the classic example is unencrypted radio communications between military units which are spied on by the enemy. In a common modern variation on the classic man-in-the-middle attack pattern, a fraudster is actually interacting with the legitimate website, and the victim is interacting with the fraudster's counterfeit website. A victim who is lured to a fraudulent website then triggers the attack by entering the normal login credentials on the counterfeit website. The counterfeit website then transmits these stolen credentials to the legitimate website using scripts or other protocols and the legitimate website then initiates a telephone call to the victim. Believing the website to be legitimate, the victim pushes the appropriate buttons on the phone, or transmits the telephoned codes to the fraudster, not realizing that doing so permits the fraudster to complete entry into the victim's account for complete access.

**Assignment to the bearer**

One basic limitation associated with relying exclusively on mobile phones for secondary authentication is the fact that the respective user must have access to a mobile phone during authentication. The user may have registered a mobile phone number, for example, and when attempting to authenticate from home, must have access to that registered mobile phone. That converts the mobile phone from an office appliance to a personal appliance for usage out of the premises. However, as soon as the mobile phone gets lost, the bearer loses physical control over the mobile authentication factors.

**SMS one time password**

SMS one time password uses information sent in an SMS to the user as part of the login process. One scenario is where a user either registers (or updates) their contact information on a website. During this time the user is also asked to enter his or her regularly used telephone numbers (home, mobile, work, etc.). The next time the user logs into the website, they must enter their username and password; if they enter the correct information, the user then chooses the phone number at which they can be contacted immediately from their
previously registered phone numbers. The user will be instantly called or receive an SMS text message with a unique, temporary PIN code. The user then enters this code into the website to prove their identity, and if the PIN code entered is correct, the user will be granted access to their account. This process provides an extra layer of online security beyond merely a username and password. These solutions can be used with any telephone, not just mobile devices.

As with any out-of-band authentication method, SMS one time password (OTP) methods are also vulnerable to man-in-the-middle (MITM) attacks because the victim, unaware of the intruder, willingly enters the temporary pin to the website, unwittingly providing the intruder access to their account. An alternative approach to OTP used primarily in Finland, is to assign a pseudo-random reply number from a pool of numbers to the communication and authenticate the user entirely out of band using that reply address along with the originating customer number and reply options within the body of the message to authenticate the user. This approach minimizes vulnerability to MITM attacks by asking the user to verify the transaction details within the body of the text message as well as on-line if they don't match, the user can see plainly that there is a MITM attack in progress and reject the transaction. It's important to note that any SMS solution may be vulnerable to mobile number porting attacks. In this scenario, an attacker tricks a mobile provider into transferring a victim's mobile number to a new account under the attacker's control. Any SMS messages or calls sent to the victim's mobile number will instead be sent only to the attacker. The victim may be unaware of the attack until the victim notices their cell phone is no longer working, or is no longer assigned the same mobile number.

**Smartphone push**

The push notification services offered by modern mobile platforms, such as iPhone's APNS and Android's C2DM/GCM, can be used to provide a real-time challenge/response mechanism on a mobile device. Upon performing a sensitive transaction or login, the user will instantly receive a challenge pushed to their mobile phone, be prompted with the full details of that transaction, and be able to respond to approve or deny that transaction by simply pressing a button on their mobile phone. Smartphone push two-factor authentication has the capability to not only be more user-friendly, but also more secure as a mutually-authenticated connection can be established to the phone over the data network.
**Additional phone token**

There is a newer method of using the mobile phone as the processor and having the Security Token reside on the mobile as a Java ME client. This method does not include data latency or incur hidden costs for the end user. While this method can simplify deployment, reduce logistical costs, and remove the need for a separate hardware token devices, there are numerous trade-offs.

Users incur fees for text/data services or cellular calling minutes. In addition, there is a variable latency involved with SMS services, especially during peak SMS usage periods like holidays. Finally, as with telephone-based processes, these processes are also vulnerable to MITM attacks, such as a victim unwittingly supplying login credentials to a counterfeit website. The counterfeit website passes these to the legitimate website using scripts or other protocols. The legitimate website then initiates an SMS text message delivery of a one-time-password to the victim's mobile device or simply waits for the Java token value to be generated. The victim enters the one-time-password onto the counterfeit website, which then forwards this to the legitimate website, where the waiting fraudster uses it to complete the fraudulent access.

**Mobile signature**

Mobile signatures are digital signatures created on a SIM card securely on a mobile device by a user's private key. In such a system text to be signed is securely sent to the SIM card on a mobile phone. The SIM then displays the text to the end-user who checks it before entering a PIN code to create a signature which is then sent back to the service provider. The signature can be verified using standard PKI systems.

Mobile Signature systems have been in use for several years. However, as with magnetic card and client digital certificate solutions, they are vulnerable to malware, are costly to deploy and support, and are strongly resisted by consumers.

**Mobile applications**

Smart phones and tablets can use a dedicated mobile device application for secure access to online services. The mobile device application uses the Web browser or Web service capabilities of the device for authentication and subsequent access to the service. This
approach allows a cryptographic key to be used to authenticate the user, which protects against a man-in-the-middle attack.

2.2.3 Inherence factors: "something only the user is"

**Biometrics**

Biometric authentication also satisfies the regulatory definition of true multi-factor authentication. Users may biometrically authenticate via their fingerprint, voiceprint, or iris scan using provided hardware and then enter a PIN or password in order to open the credential vault. However, while this type of authentication is suitable in limited applications, this solution may become unacceptably slow and comparatively expensive when a large number of users are involved. In addition, it is extremely vulnerable to a replay attack: once the biometric information is compromised, it may easily be replayed unless the reader is completely secure and guarded. Voice Biometrics has the distinct advantage that it can ask the user to say random phrases, and thereby significantly reduce the risk of a successful replay attack. Finally, there is great user resistance to biometric authentication. Users resist having their personal physical characteristics captured and recorded for authentication purposes. In short, selection and successful deployment of a biometric authentication system needs careful consideration of many factors.

For many biometric identifiers, the actual biometric information is rendered into string or mathematic information. The device scans the physical characteristic, extracts critical information, and then stores the result as a string of data. Comparison is therefore made between two data strings, and if there is sufficient commonality a pass is achieved. It may be appreciated that choice of how much data to match, and to what degree of accuracy, governs the accuracy/speed ratio of the biometric device. All biometric devices, therefore, do not provide unambiguous guarantees of identity, but rather probabilities and all may provide false positive and negative outputs. If a biometric system is applied to a large number of users - perhaps all of the customers of a bank - the error rate may make the system impractical to use.

Biometric information may be mechanically copied and cannot be easily changed. This is perceived as a key disadvantage since, if discovered, the compromised data cannot be changed. A user can easily change his/her password; however, a user cannot change their fingerprint. A bio-identifier can also be faked. For example, fingerprints can be captured on
sticky tape and false gelatine copies made, or simple photos of eye retinas can be presented. More expensive biometrics sensors should be capable to distinguish between live original and dead replicas, but such devices are not practical for mass distribution. It is likely that, as biometric identifiers become widespread, more sophisticated compromise techniques will also be developed.

Historically, fingerprints have been used as the most authoritative method of authentication. Other biometric methods such as retinal scans are promising, but have shown themselves to be easily spoofable in practice. Hybrid or two-tiered authentication methods offer a compelling solution, such as private keys encrypted by fingerprint inside of a USB device.

A criticism of biometrics for authentication is that whereas it is relatively easy to calculate the strength of a password from its length and composition and hence the time to brute force it, the strength of a biometric is difficult to quantify. There can be no guarantee that a simple attack could not be devised tomorrow, for example by using household chemicals to make an artificial finger from a fingerprint, good enough to be accepted by a fingerprint reader. This is a concern to certain government security authorities where knowing the strength of a security mechanism is considered more important than having a mechanism which might be stronger but whose absolute strength is not quantifiable.

International travelers to many countries are now routinely required to provide fingerprint and/or iris scans to pass. This stockpile of records reduces the strength of biometric-protected resources (for example - those governments can now potentially unlock your PC, decrypt your flash drives, and/or impersonate you, although implementation may be non-trivial).

2.2.4 Cost effectiveness
There are drawbacks to two-factor authentication that are keeping many approaches from becoming widespread. Some consumers have difficulty keeping track of a hardware token or USB plug. Many consumers do not have the technical skills needed to install a client-side software certificate.

As a result, adding a second factor to the authentication process typically leads to increase in costs for implementation and maintenance. Most hardware token-based systems are proprietary and charge an annual fee per user in the $50–100 USD range. Deployment of
hardware tokens is logistically challenging. Hardware tokens may get damaged or lost and issuance of tokens in large industries such as banking or even within large enterprises needs to be managed.

In addition to deployment costs, two-factor authentication often carries significant additional support costs. A 2008 survey of over 120 U.S. credit unions by the *Credit Union Journal* reported on the support costs associated with two-factor authentication. In their report, software certificates and software toolbar approaches were reported to have the highest support costs.

### 2.2.5 Market acceptance

As a result of challenges with integration and user acceptance, true two-factor authentication is not yet widespread, although it can be found in certain sectors requiring additional security (e.g. banking, military). Faced with regulatory two-factor authentication guidelines in 2005, numerous U.S. financial institutions instead deployed additional knowledge-based authentication methods, such as shared secrets or challenge questions, only to discover later that such methods do not satisfy the regulatory definition of "true multifactor authentication". Supplemental regulatory guidelines and stricter enforcement are now beginning to force the abandonment of knowledge-based methods in favor of "true multifactor authentication".

A 2007 study sponsored by BearingPoint reported 94% of the authentication solutions implemented by U.S. financial institutions fail to meet the regulatory definition of true multifactor authentication.

An increasing count of recent undesired disclosure of governmentally protected data or private date is likely to contribute to new TF-A requirements, especially in the European Union.

### 2.2.6 Man-in-the-browser

**Man-in-the-browser (MITB, MitB, MIB, MiB)**, a form of Internet threat related to man-in-the-middle (MITM), is a proxy Trojan horse that infects a web browser by taking advantage of vulnerabilities in browser security to modify web pages, modify transaction content or insert additional transactions, all in a completely covert fashion invisible to both the user and host web application. A MitB attack will be successful irrespective of whether security
mechanisms such as SSL/PKI and/or two or three-factor Authentication solutions are in place. A MitB attack may be countered by utilising out-of-band transaction verification, although SMS verification can be defeated by man-in-the-mobile (MitMo) malware infection on the mobile phone. Trojans may be detected and removed by antivirus software with a 23% success rate against Zeus in 2009, and still low rates in 2011. The 2011 report concluded that additional measures on top of antivirus were needed. A related, simpler attack is the boy-in-the-browser (BitB, BITB). The majority of financial service professionals in a survey considered MitB to be the greatest threat to online banking.

2.2.7 Man-in-the-middle attack
The man-in-the-middle attack (often abbreviated MITM, MitM, MIM, MiM, MITMA) in cryptography and computer security is a form of active eavesdropping in which the attacker makes independent connections with the victims and relays messages between them, making them believe that they are talking directly to each other over a private connection, when in fact the entire conversation is controlled by the attacker. The attacker must be able to intercept all messages going between the two victims and inject new ones, which is straightforward in many circumstances (for example, an attacker within reception range of an unencrypted Wi-Fi wireless access point, can insert himself as a man-in-the-middle).

A man-in-the-middle attack can succeed only when the attacker can impersonate each endpoint to the satisfaction of the other & it is an attack on mutual authentication (or lack thereof). Most cryptographic protocols include some form of endpoint authentication specifically to prevent MITM attacks. For example, SSL can authenticate one or both parties using a mutually trusted certification authority.

2.2.8 One Time Password
Traditional hardware tokens, SMS, and telephone-based methods are vulnerable to a type of attack known as the man-in-the-middle, or MITM attack because they are physically disconnected from the authenticating entity. In such an attack a fraudster impersonates a bank or similar authenticating entity to the customer, prompting the victim to divulge to them the value generated by their token or other authentication process. The fraudster then passes this (valid) authentication factor to the genuine bank or similar authenticating entity instead of the user. The fraudster does not need to be in physical possession of the hardware token, SMS device, or telephone to compromise the victim's account. They only need to solicit the
authentication information and then pass it on to the genuine website within the appropriate time frame. Citibank made headline news in 2006 when its hardware token-equipped business customers were targeted by just such an attack from fraudsters based in the Ukraine. Such an attack may be used to gain information about the victim’s accounts, or to get them to authorize a transfer of a different sum to a different recipient than intended.

One Factor Authentication risks can be greatly reduced by continually altering the password using Two-Factor Authentication mechanism e.g. RSA dongles, biometrics (iris scan and finger print readers). However these devices are not portable hence are greatly challenged by cloud computing technology which is increasingly being accessed by smart devices e.g. tablets. Physical sizes of mobile devices limit the number of peripherals and ports which could be inbuilt into them; cloud technology cannot be supported by the onsite devices mentioned above. In addition to the above constraint, mobile devices (due to mobility) often experience fluctuating, limited internet connectivity hence not suitable for bandwidth-intensive authentication models like finger print reader.

Mobile One Time Password (Mobile OTP) has addressed portability issue, user interface, less bandwidth, quick service delivery among others. However, Mobile OTP also has challenges namely: Social Engineering, OTP Profile Cloning using malware and phishing attacks.
2.3 M-Pesa Integration

2.3.1 Traditional E-Commerce

**Electronic commerce**, commonly known as **e-commerce** or **eCommerce**, is a type of industry where the buying and selling of products or services is conducted over electronic systems such as the Internet and other computer networks. Electronic commerce draws on technologies such as mobile commerce, electronic funds transfer, supply chain management, Internet marketing, online transaction processing, electronic data interchange (EDI), inventory management systems, and automated data collection systems. Modern electronic commerce typically uses the World Wide Web at least at one point in the transaction's lifecycle, although it may encompass a wider range of technologies such as e-mail, mobile devices, social media, and telephones as well.

Electronic commerce is generally considered to be the sales aspect of e-business. It also consists of the exchange of data to facilitate the financing and payment aspects of business transactions. This is an effective and efficient way of communicating within an organization and one of the most effective and useful ways of conducting business.

E-commerce can be divided into:

- E-tailing or "virtual storefronts" on websites with online catalogs, sometimes gathered into a "virtual mall"
- Buying or Selling on various websites and/or online marketplaces
- The gathering and use of demographic data through Web contacts and social media
- Electronic Data Interchange (EDI), the business-to-business exchange of data
- E-mail and fax and their use as media for reaching prospective and established customers (for example, with newsletters)
- Business-to-business buying and selling
- The security of business transactions

2.3.2 Mobile payment

**Mobile payment**, also referred to as mobile money, mobile money transfer, and mobile wallet generally refer to payment services operated under financial regulation and performed from or via a mobile device. Instead of paying with cash, cheque, or credit cards, a consumer can use a mobile phone to pay for a wide range of services and digital or hard goods.
Although the concept of using non-coin-based currency systems has a long history, it is only recently that the technology to support such systems has become widely available.

Mobile payment is being adopted all over the world in different ways. In 2008, the combined market for all types of mobile payments was projected to reach more than $600B globally by 2013, which would be double the figure as of February, 2011. The mobile payment market for goods and services, excluding contactless Near Field Communication or NFC transactions and money transfers, is expected to exceed $300B globally by 2013.

In developing countries mobile payment solutions have been deployed as a means of extending financial services to the community known as the "unbanked" or "underbanked," which is estimated to be as much as 50% of the world's adult population, according to Financial Access’ 2009 Report "Half the World is Unbanked". These payment networks are often used for micropayments. The use of mobile payments in developing countries has attracted public and private funding by organizations such as the Bill and Melinda Gates Foundation, USAID and MercyCorps.

**Models**

There are four primary models for mobile payments:

- Premium SMS based transactional payments
- Direct Mobile Billing
- Mobile Web Payments (WAP)
- Contactless NFC (Near Field Communication)

Financial institutions and credit card companies as well as Internet companies such as Google and a number of mobile communication companies, such as mobile network operators and major telecommunications infrastructure such as w-HA from Orange and handset multinationals such as Ericsson have implemented mobile payment solutions.

**SMS/USSD-based transactional payments**

The consumer sends a payment request via an SMS text message or an USSD to a short code and a premium charge is applied to their phone bill or their online wallet. The merchant involved is informed of the payment success and can then release the paid for goods.
Since a trusted delivery address has typically not been given these goods are most frequently digital with the merchant replying using a Multimedia Messaging Service to deliver the purchased music, ringtones, wallpapers etc.

A Multimedia Messaging Service can also deliver barcodes which can then be scanned for confirmation of payment by a merchant. This is used as an electronic ticket for access to cinemas and events or to collect hard goods.

Transactional payments have been popular in Asia and Europe but are now being overtaken by other mobile payment methods, such as mobile web payments (WAP), mobile payment client (Java ME, Android...) and Direct Mobile Billing.

Possible reasons include:

Poor reliability - transactional payments can easily fail as messages get lost.

1. Slow speed - sending messages can be slow and it can take hours for a merchant to get receipt of payment. Consumers do not want to be kept waiting more than a few seconds.

2. Security - The SMS/USSD encryption ends in the radio interface, then the message is a plaintext.

3. High cost - There are many high costs associated with this method of payment. The cost of setting up short codes and paying for the delivery of media via a Multimedia Messaging Service and the resulting customer support costs to account for the number of messages that get lost or are delayed.

4. Low payout rates - operators also see high costs in running and supporting transactional payments which results in payout rates to the merchant being as low as 30%. Usually around 50%

5. Low follow-on sales - once the payment message has been sent and the goods received there is little else the consumer can do. It is difficult for them to remember where something was purchased or how to buy it again. This also makes it difficult to tell a friend.
Some mobile payment services accept "premium SMS payments." Here is the typical end user payment process:

1. User sends SMS with keyword and unique number to a premium short code.
2. User receives a PIN (User billed via the short code on receipt of the PIN)
3. User uses PIN to access content or services.

Direct Mobile Billing
The consumer uses the mobile billing option during checkout at an e-commerce site such as an online gaming site to make a payment. After two-factor authentication involving a PIN and One-Time-Password (often abbreviated as OTP), the consumer's mobile account is charged for the purchase. It is a true alternative payment method that does not require the use of credit/debit cards or pre-registration at an online payment solution such as PayPal, thus bypassing banks and credit card companies altogether. This type of mobile payment method, which is extremely prevalent and popular in Asia, provides the following benefits:

2. Convenience - No pre-registration and no new mobile software is required.
3. Easy - It's just another option during the checkout process.
4. Fast - Most transactions are completed in less than 10 seconds.
5. Proven - 70% of all digital content purchased online in some parts of Asia uses the Direct Mobile Billing method

Mobile Web Payments (WAP)
The consumer uses web pages displayed or additional applications downloaded and installed on the mobile phone to make a payment. It uses WAP (Wireless Application Protocol) as underlying technology and thus inherits all the advantages and disadvantages of WAP. Benefits include:

1. Follow-on sales where the mobile web payment can lead back to a store or to other goods the consumer may like. These pages have a URL and can be bookmarked making it easy to re-visit or share.
2. High customer satisfaction from quick and predictable payments
3. Ease of use from a familiar set of online payment pages
However, unless the mobile account is directly charged through a mobile network operator, the use of a credit/debit card or pre-registration at online payment solution such as PayPal is still required just as in a desktop environment.

Mobile web payment methods are now being mandated by a number of mobile network operators.

**Direct Operator Billing**

Direct operator billing, also known as mobile content billing, WAP billing, and carrier billing, requires integration with the operator. It provides certain benefits:

1. The operators already have a billing relationship with the consumers; the payment will be added to their bill.
2. Provides instantaneous payment
3. Protect payment details and consumer identity
4. Better conversion rates
5. Reduced customer support costs *for merchants*

One drawback: the payout rate will be much lower than with other payment providers.

Examples from a popular provider:

- 92% with Paypal
- 85 to 86% with Credit Card
- 45 to 91.7% with operator billing in the US, UK and some smaller European countries, but usually around 60%

More recently, Direct Operator Billing is being deployed in an in-app environment, where mobile application developers are taking advantage of the one-click payment option that Direct Operator Billing provides for monetising mobile applications. This is a logical alternative to credit card and Premium SMS billing.

In 2012, Ericsson and Western Union partnered to expand the direct operator billing market, making it possible for mobile operators to include Western Union Mobile Money Transfers as part of their mobile financial service offerings. Given the international reach of both companies, the partnership is meant to accelerate the interconnection between the m-commerce market and the existing financial world.
**Credit Card**
A simple mobile web payment system can also include a credit card payment flow allowing a consumer to enter their card details to make purchases. This process is familiar but any entry of details on a mobile phone is known to reduce the success rate (conversion) of payments. In addition, if the payment vendor can automatically and securely identify customers then card details can be recalled for future purchases turning credit card payments into simple single click-to-buy giving higher conversion rates for additional purchases.

**Online Wallets**
Online companies like PayPal, Amazon Payments, and Google Wallet also have mobile options.
First payment:
- User registers, inputs their phone number, and the provider sends them an SMS with a PIN
- User enters the received PIN, authenticating the number
- User inputs their credit card info or another payment method if necessary (not necessary if the account has already been added) and validates payment
Subsequent payments:
- The user re enters their PIN to authenticate and validates payment
Requesting a PIN is known to lower the success rate (conversion) for payments. These systems can be integrated with directly or can be combined with operator and credit card payments through a unified mobile web payment platform.

**Contactless Near Field Communication**
Near Field Communication (NFC) is used mostly in paying for purchases made in physical stores or transportation services. A consumer using a special mobile phone equipped with a smartcard waves his/her phone near a reader module. Most transactions do not require authentication, but some require authentication using PIN, before transaction is completed. The payment could be deducted from a pre-paid account or charged to a mobile or bank account directly.
Mobile payment method via NFC faces significant challenges for wide and fast adoption, due to lack of supporting infrastructure, complex ecosystem of stakeholders, and standards. Some phone manufacturers and banks, however, are enthusiastic. Ericsson and Aconite are
examples of businesses that make it possible for banks to create consumer mobile payment applications that take advantage of NFC technology.

NFC vendors in Japan are closely related to mass-transit networks, like the Mobile Suica used on the JR East rail network. Osaifu-Keitai system, used for Mobile Suica and many others including Edy and nanaco, has become the de facto standard method for mobile payments in Japan. Its core technology, Mobile FeliCa IC, is partially owned by Sony, NTT DoCoMo and JR East. Mobile FeliCa utilize Sony's FeliCa technology, which itself is the de facto standard for contactless smart cards in the country.

Other NFC vendors mostly in Europe use contactless payment over mobile phones to pay for on- and off-street parking in specially demarcated areas. Parking wardens may enforce the parkings by license plate, transponder tags or barcode stickers. First conceptualized in the 1990s the technology has seen commercial use in this century in both Scandinavia and Estonia. End users benefit from the convenience of being able to pay for parking from the comfort of their car with their mobile phone, and parking operators are not obliged to invest in either existing or new street-based parking infrastructures. Parking wardens maintain order in these systems by license plate, transponder tags or barcode stickers or they read a digital display in the same way as they read a pay and display receipt.

Other vendors use a combination of both NFC and a barcode on the mobile device for mobile payment, for example, Cimbal or DigiMo, making this technique attractive at the point of sale because many mobile devices in the market do not yet support NFC.

Cloud-based mobile payments
Google, PayPal, GlobalPay and GoPago use a cloud-based approach to in-store mobile payment. The cloud based approach places the mobile payment provider in the middle of the transaction, which involves two separate steps. First, a cloud-linked payment method is selected and payment is authorized via NFC or an alternative method. During this step, the payment provider automatically covers the cost of the purchase with issuer linked funds. Second, in a separate transaction, the payment provider charges the purchaser's selected, cloud-linked account in a card-not-present environment to recoup its losses on the first transaction.
Audio signal-based payments
The audio channel of the mobile phone is another wireless interface that is used to make payments. Several companies have created technology to use the acoustic features of cell phones to support mobile payments and other applications that are not chip-based. The technologies Near sound data transfer (NSDT), Data Over Voice and NFC 2.0 produce audio signatures that the microphone of the cell phone can pick up to enable electronic transactions.

Direct carrier/bank co-operation
In the T-Cash model the mobile phone and the phone carrier is the front end interface to the consumers. The consumer can purchase goods, transfer money to a peer, cash-out, and cash-in. A 'mini wallet' account can be opened as simply as entering *700# on the mobile phone, presumably by depositing money at a participating local merchant and the mobile phone number. Presumably other transactions are similarly accomplished by entering special codes and the phone number of the other party on the consumer's mobile phone.

Mobile payment service provider model
There are four potential mobile payment models:

1. *Operator-Centric Model*: The mobile operator acts independently to deploy mobile payment service. The operator could provide an independent mobile wallet from the user mobile account (airtime). A large deployment of the Operator-Centric Model is severely challenged by the lack of connection to existing payment networks. Mobile network operator should handle the interfacing with the banking network to provide advanced mobile payment service in banked and under banked environment. Pilots using this model have been launched in emerging countries but they did not cover most of the mobile payment service use cases. Payments were limited to remittance and airtime top up.

2. *Bank-Centric Model*: A bank deploys mobile payment applications or devices to customers and ensures merchants have the required point-of-sale (POS) acceptance capability. Mobile network operator are used as a simple carrier, they bring their experience to provide Quality of service (QOS) assurance.

3. *Collaboration Model*: This model involves collaboration among banks, mobile operators and a trusted third party.
4. **Peer-to-Peer Model**: The mobile payment service provider acts independently from financial institutions and mobile network operators to provide mobile payment. For example the MHITS SMS payment service uses a peer-to-peer model.

### 2.3.3 M-Pesa Introduction

**M-Pesa** (M for mobile, *pesa* is Swahili for money) is a mobile-phone based money transfer and micro-financing service for Safaricom and Vodacom, the largest mobile network operators in Kenya and Tanzania. Currently the most developed mobile payment system in the world, M-Pesa allows users with a national ID card or passport to deposit, withdraw, and transfer money easily with a mobile device.

### 2.3.4 M-Pesa Concept

The initial concept of M-Pesa was to create a service which allowed microfinance borrowers to conveniently receive and repay loans using the network of Safaricom airtime resellers. This would enable microfinance institutions (MFIs) to offer more competitive loan rates to their users, as costs are lower than when dealing in cash. The users of the service would gain through being able to track their finances more easily. When the service was piloted, customers adopted the service for a variety of alternative uses and complications arose with Faulu, the partnering MFI. In discussion with other parties, M-Pesa was re-focused and launched with a different value proposition: sending remittances home across the country and making payments.

M-Pesa is a branchless banking service, meaning that it is designed to enable users to complete basic banking transactions without visiting a bank branch. The continuing success of M-Pesa in Kenya has been due to the creation of a highly popular, affordable payment service with only limited involvement of a bank. M-Pesa is also criticized for robbing the government from seigniorage revenues.

### 2.3.5 M-Pesa Services

M-Pesa customers can deposit and withdraw money from a network of agents that includes airtime resellers and retail outlets acting as banking agents. M-Pesa is operated by Safaricom, a mobile network operator (MNO), which is not classed as a deposit-taking institution (such as a bank).
The service enables its users to:

- Deposit and withdraw money
- Transfer money to other users and non-users
- Pay bills
- Purchase airtime
- Transfer money between the service and a bank account (in some markets)

The user interface technology of M-Pesa differs between Safaricom of Kenya and Vodacom of Tanzania, although the underlying platform is the same. While Safaricom uses SIM toolkit (STK) to provide handset menus for accessing the service, Vodacom relies mostly on USSD to provide users with menus, but also supports STK.

2.3.6 M-Pesa Integration Challenges

Despite the historical success of the M-Pesa Business Model, there is an outcry over a published Application Porting Interface (API) for integration of the M-Pesa system with the online solutions. This hinders the growth of the product which commands a large subscriber base.

2.3.7 M-Pesa Integration Justification

Statistics shows that M-Pesa service counts for over 40% of all money transferred within the country. With an exponentially growing subscriber base of 8 million in 2010, the M-Pesa traffic is still attracting more users and revenue. CCK Mobile Penetration Statistics is placing the penetration at slightly above 75% by March 2013. The statistics justify the need to develop an e-commerce model for Mobile Money Payment Platforms so as to add value to the service which has been widely accepted.
2.4 Summary
It is necessary to mitigate some of the Mobile OTP weaknesses by providing an alternative Multi-Factor Authentication mechanism (which is widely accepted in the cloud computing technology) so as to reduce risks associated with MOTP Authentication.

With the wide acceptance of Mobile Money Payment System and subsequent exponential growth of the same in the local market, an alternative E-commerce model needs to be developed to take advantage of the same.

Online solutions that are increasingly accessed using mobile devices must be secured against all forms of risks including authentication among others. More so, the solutions require a medium of receiving money for goods and services offered hence the necessity of integration.
Chapter 3: Methodology

3.1 Introduction
According to Wikipedia, a Software Development Methodology or System Development Methodology in Software Engineering is a framework that is used to structure, plan and control the process of developing an information system. Common methodologies include waterfall, prototyping, iterative and incremental development, spiral development, rapid application development and extreme programming. A methodology can also include aspects of the development environment (i.e. IDEs), model-based development, computer aided software development, and the utilization of particular frameworks (i.e. programming libraries or other tools).

3.2 Choice of Model
For the purpose of this Research, we shall focus more on the Software Prototyping Model. Occasionally, we shall borrow a few steps of the Waterfall Model to help implement further the prototype in order to create visibility of the project.

3.3 Waterfall Model
The waterfall model is a sequential development approach, in which development is seen as flowing steadily downwards (like a waterfall) through the phases of requirements analysis, design, implementation, testing (validation), integration and maintenance. The first formal description of the method is often cited as an article published by Winston W. Royce in 1970 although Royce did not use the term "waterfall" in this article.
The basic principles are:-

- Project is divided into sequential phases, with some overlap and splashback acceptable between phases.
- Emphasis is on planning, time schedules, target dates, budgets and implementation of an entire system at one time.
- Tight control is maintained over the life of the project via extensive written documentation, formal reviews, and approval/signoff by the user and information technology management occurring at the end of most phases before beginning the next phase.

The Waterfall model is a traditional engineering approach applied to software engineering. It has been widely blamed for several large-scale government projects running over budget, over time and sometimes failing to deliver on requirements due to the Big Design Up Front approach. Except when contractually required, the Waterfall model has been largely superseded by more flexible and versatile methodologies developed specifically for software development.
Advantages of waterfall model

- Simple and easy to understand and use
- Easy to manage due to the rigidity of the model — each phase has specific deliverables and a review process
- Phases are processed and completed one at a time
- Works well for smaller projects where requirements are very well understood

Disadvantages of waterfall model

- Once an application is in the testing stage, it is very difficult to go back and change something that was not well-thought out in the concept stage
- No working software is produced until late during the life cycle
- High amounts of risk and uncertainty
- Not a good model for complex and object-oriented projects
- Poor model for long and ongoing projects
- Not suitable for the projects where requirements are at a moderate to high risk of changing

When to use the waterfall model

- Requirements are very well known, clear and fixed
- Product definition is stable
- Technology is understood
- There are no ambiguous requirements
- Ample resources with required expertise are available freely
- The project is short
3.4 Software Prototyping

**Software prototyping** refers to the activity of creating prototypes of software applications, i.e., incomplete versions of the software program being developed. It is an activity that can occur in software development and is comparable to prototyping as known from other fields, such as mechanical engineering or manufacturing. A prototype typically simulates only a few aspects of, and may be completely different from, the final product.

![Software Prototyping Diagram](image)

**Fig. 3 - Software Prototyping**

3.4.1 Software Prototyping Steps

1. **Identify basic requirements**
   Determine basic requirements including the input and output information desired. Details such as security can typically be ignored.

2. **Develop Initial Prototype**
   The initial prototype is developed, focusing on user interfaces.

3. **Review**
   The customers, including end-users, examine the prototype and provide feedback on additions or changes.

4. **Revise and Enhance the Prototype**
   Using the feedback both the specifications and the prototype can be improved. Negotiation about what is within the scope of the contract/product may be necessary. If changes are introduced then a repeat of steps #3 and #4 may be needed.
3.4.2 Dimensions of prototypes

Nielsen summarizes the various dimensions of prototypes in his book Usability Engineering.

**Horizontal Prototype**

A common term for a user interface prototype is the **horizontal prototype**. It provides a broad view of an entire system or subsystem, focusing on user interaction more than low-level system functionality, such as database access. Horizontal prototypes are useful for:

- Confirmation of user interface requirements and system scope
- Demonstration version of the system to obtain buy-in from the business
- Develop preliminary estimates of development time, cost and effort.

**Vertical Prototype**

A **vertical prototype** is a more complete elaboration of a single subsystem or function. It is useful for obtaining detailed requirements for a given function, with the following benefits:

- Refinement database design
- Obtain information on data volumes and system interface needs, for network sizing and performance engineering
- Clarifies complex requirements by drilling down to actual system functionality

3.4.3 Types of prototyping

Software prototyping has many variants. However, all the methods are in some way based on two major types of prototyping: Throwaway Prototyping and Evolutionary Prototyping.

**Throwaway prototyping**

Also called close-ended prototyping, Throwaway or Rapid Prototyping refers to the creation of a model that will eventually be discarded rather than becoming part of the final delivered software. After preliminary requirements gathering is accomplished, a simple working model of the system is constructed to visually show the users what their requirements may look like when they are implemented into a finished system. Rapid Prototyping involved creating a working model of various parts of the system at a very early stage, after a relatively short investigation. The method used in building it is usually quite informal, the most important factor being the speed with which the model is provided. The model then becomes the starting point from which users can re-examine their expectations and clarify their requirements. When this has been achieved, the prototype model is 'thrown away', and the system is formally developed based on the identified requirements.
The most obvious reason for using Throwaway Prototyping is that it can be done quickly. If the users can get quick feedback on their requirements, they may be able to refine them early in the development of the software. Making changes early in the development lifecycle is extremely cost effective since there is nothing at that point to redo. If a project is changed after a considerable work has been done then small changes could require large efforts to implement since software systems have many dependencies. Speed is crucial in implementing a throwaway prototype, since with a limited budget of time and money little can be expended on a prototype that will be discarded.

Another strength of Throwaway Prototyping is its ability to construct interfaces that the users can test. The user interface is what the user sees as the system, and by seeing it in front of them, it is much easier to grasp how the system will work. It is asserted that revolutionary rapid prototyping is a more effective manner in which to deal with user requirements-related issues, and therefore a greater enhancement to software productivity overall. Requirements can be identified, simulated, and tested far more quickly and cheaply when issues of evolvability, maintainability, and software structure are ignored. This, in turn, leads to the accurate specification of requirements and the subsequent construction of a valid and usable system from the user's perspective via conventional software development models.

Prototypes can be classified according to the fidelity with which they resemble the actual product in terms of appearance, interaction and timing. One method of creating a low fidelity Throwaway Prototype is Paper Prototyping. The prototype is implemented using paper and pencil, and thus mimics the function of the actual product, but does not look at all like it. Another method to easily build high fidelity Throwaway Prototypes is to use a GUI Builder and create a click dummy, a prototype that looks like the goal system, but does not provide any functionality.

Not exactly the same as Throwaway Prototyping, but certainly in the same family, is the usage of storyboards, animatics or drawings. These are non-functional implementations but show how the system will look.
In this approach the prototype is constructed with the idea that it will be discarded and the final system will be built from scratch. The steps in this approach are:

1. Write preliminary requirements
2. Design the prototype
3. User experiences/uses the prototype, specifies new requirements
4. Repeat if necessary
5. Write the final requirements
6. Develop the real products

**Evolutionary prototyping**

Evolutionary Prototyping (also known as breadboard prototyping) is quite different from Throwaway Prototyping. The main goal when using Evolutionary Prototyping is to build a very robust prototype in a structured manner and constantly refine it. The reason for this is that the Evolutionary prototype, when built, forms the heart of the new system, and the improvements and further requirements will be built.

When developing a system using Evolutionary Prototyping, the system is continually refined and rebuilt.

"...evolutionary prototyping acknowledges that we do not understand all the requirements and builds only those that are well understood."

This technique allows the development team to add features, or make changes that couldn't be conceived during the requirements and design phase. For a system to be useful, it must evolve through use in its intended operational environment. A product is never "done;" it is always maturing as the usage environment changes. We often try to define a system using our most familiar frame of reference---where we are now. We make assumptions about the way business will be conducted and the technology base on which the business will be implemented. A plan is enacted to develop the capability, and, sooner or later, something resembling the envisioned system is delivered.

Evolutionary Prototypes have an advantage over Throwaway Prototypes in that they are functional systems. Although they may not have all the features the users have planned, they may be used on an interim basis until the final system is delivered.
"It is not unusual within a prototyping environment for the user to put an initial prototype to practical use while waiting for a more developed version. The user may decide that a 'flawed' system is better than no system at all.

In Evolutionary Prototyping, developers can focus themselves to develop parts of the system that they understand instead of working on developing a whole system. To minimize risk, the developer does not implement poorly understood features. The partial system is sent to customer sites. As users work with the system, they detect opportunities for new features and give requests for these features to developers. Developers then take these enhancement requests along with their own and use sound configuration-management practices to change the software-requirements specification, update the design, recode and retest.

**Incremental prototyping**
The final product is built as separate prototypes. At the end the separate prototypes are merged in an overall design.

**Extreme prototyping**
Extreme Prototyping as a development process is used especially for developing web applications. Basically, it breaks down web development into three phases, each one based on the preceding one. The first phase is a static prototype that consists mainly of HTML pages. In the second phase, the screens are programmed and fully functional using a simulated services layer. In the third phase, the services are implemented. The process is called Extreme Prototyping to draw attention to the second phase of the process, where a fully functional UI is developed with very little regard to the services other than their contract.

### 3.4.4 Advantages of Prototyping
There are many advantages to using prototyping in software development - some tangible, some abstract.

**Reduced time and costs:** Prototyping can improve the quality of requirements and specifications provided to developers. Because changes cost exponentially more to implement as they are detected later in development, the early determination of what the user really wants can result in faster and less expensive software.
Improved and increased user involvement: Prototyping requires user involvement and allows them to see and interact with a prototype allowing them to provide better and more complete feedback and specifications. The presence of the prototype being examined by the user prevents many misunderstandings and miscommunications that occur when each side believe the other understands what they said. Since users know the problem domain better than anyone on the development team does, increased interaction can result in final product that has greater tangible and intangible quality. The final product is more likely to satisfy the users’ desire for look, feel and performance.

3.4.5 Prototyping Summary
The research will incline more on the Incremental software Prototyping Method. This due to limited User and System Requirement Specifications and time constrains. Additional functionality shall be added on need basis.

3.5 Design and Implementation
3.5.1 Tools and Requirements
There are different technologies that could be used to prototype the project. However, our focus shall be on widely used and free (Open Source) technologies so as to minimize the Cost of the Project and Ownership (licensing and subscription).

Operating System
- Linux Platform e.g. CentOS (kernel version 2.6++)

Relational Database Management
- MySQL (version 5++)

Programming Language
- PHP (version 5++)

Integration Gateways
- SMTP (for e-mail notifications)
- SMPP or Bulk SMS API (for SMS notifications)
Other Requirements

- M-Pesa Paybill Account (e.g. 881400) with Instant Payment Notification (IPN) integration
- M-Pesa User Account with Units
- Internet Access

Testing Environment

- Standard Web browser e.g. Mozilla Firefox (for Testing Random Password Authentication)
- Smart Phone (with M-Pesa User Account and Client, M-OTP Client)
- SSH Client (e.g. Putty or Unix/Linux Terminal)

3.5.2 Identify basic requirements

- Create a Random Authentication Mechanism to mitigate the limitations of Mobile One Time Password Multi-Factor Authentication
  1. Ability to Generate and Store all random passwords using one way encryption
  2. Ability to adjusting random password parameters
  3. Password should be truly random with less Processing and Memory overheads
  4. Ability to Limit Login Attempts
  5. Limit Login Period to a few Seconds
  6. Attach Authentication with Randomized, Encrypted Unique Session Token and not Username (avoid backend session expiry extension by unethical system administrators)

- Securely Mobile Payment System Integration
  1. Implement Online Transactions Input Channel for Mobile Payment Notifications
  2. Create Access Control List (ACL) for source Instant Payment Notification (IPN) server Internet Protocol (IP)
  3. Authenticate IPN Source (validate username and password)
  4. Log sessions (successful and failed authentications)
  5. Send notification (e-mail and SMS) to user on progress status (successful or failed remittances)
3.5.3 Develop Initial Prototype

- **Random Password Multi-Factor Authentication**
  1. Design Schema for Random Passwords (Storage and Login Attempts)
  2. Design Sample Interface to Password Input
  3. Implement Random Password Generation, Notification (SMS, e-mail or both), Storage and Validation Logics

```php
function getRandomString($Length)
{
    $RandomString = ";
    while (strlen($RandomString) < $Length) {
        if (ctype_alnum($TempStr = chr(mt_rand(48, 122)))) {
            $RandomString .= $TempStr;
        }
    }
    return $RandomString;
}
```

4. Integrate the Design and Logic onto an Existing System (e.g. OnlineSMIS) for Testing

- **Secure Mobile Payment System Integration**
  1. Design Schema for Mobile Payment Online Transactions
  2. Design Interface for Displaying List of Transactions (i.e. Dashboard) with Processing Status (e.g. Pending, Failed)
  3. Implement and Secure Input Plugin for Mobile Payment Notifications
  4. Implement Sample Logic for Processing the Queued Notifications
  5. Implement Transaction Processing and Auto-Receipting
  6. Implement Notification Channel for Progress Status of Transactions
  7. Implement Online Transactions Dashboard with various Transaction Status
  8. Implement transaction correction utility for transactions that failed to process due to wrong account details among other errors
  9. Integrate the Design and Logic onto Existing System (e.g. OnlineSMIS) for Testing
3.5.4 Sample Schema and Flow Design Diagrams

- Random Password Multi-Factor Authentication

Fig. 4 - Random Password Authentication Flow Diagram

Table Descriptions:

**loginportals**: Portals e.g. Admin, Student and Parent

**randomsessiontypes**: Random Session States e.g. New, Authenticated, Rejected and
Fig. 5 - Random Password Authentication EER Diagram

Fig. 6 - Multi-Factor Authentication Process Flow Diagram

Fig. 7 - Random Password Authentication Dialog Box
Secure Mobile Payment System Integration

Fig. 8 - Mobile Payment System Integration Flow Diagram

Table Descriptions:

paymentaccounttypes: E.g. Local (processed locally) or Transit (forwarded to another host for processing)

paymentgateways: Configured Payment Gateways e.g. Safaricom M-Pesa, AirTel Kenya Money, Paypall etc

paymentsources: EFT (from Commercial Banks), E-Commerce (from Paypall..) or Mobile
Payment Platforms

**status:** Account State e.g. Active, Inactive or Terminated

**transactionerrorcodes:** Different Error Codes of Online Transaction Processing e.g. In-Progress, Successful etc

**Fig. 9 - Mobile Payment System Integration Entity Relationship Diagram**
Chapter 4: Results and Discussions

4.1 Plotting Performance using Kernel Density Estimation

In statistics, kernel density estimation (KDE) is a non-parametric way to estimate the probability density function of a random variable. Kernel density estimation is a fundamental data smoothing problem where inferences about the population are made, based on a finite data sample.

Let \((x_1, x_2, \ldots, x_n)\) be an independent and identically distributed sample drawn from some distribution with an unknown density \(f\). We are interested in estimating the shape of this function \(f\). Its kernel density estimator is

\[
\hat{f}_h(x) = \frac{1}{n} \sum_{i=1}^{n} K_h(x - x_i) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right),
\]

where \(K(A)\) is the kernel \(A\) a symmetric but not necessarily positive function that integrates to one \(A\) and \(h > 0\) is a smoothing parameter called the bandwidth. A kernel with subscript \(h\) is called the scaled kernel and defined as \(K_h(x) = 1/h K(x/h)\). Intuitively one wants to choose \(h\) as small as the data allow, however there is always a trade-off between the bias of the estimator and its variance; more on the choice of bandwidth below.
## 4.2 Testing Scenarios

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Random Password Multi-Factor Authentication</strong></td>
<td></td>
</tr>
<tr>
<td>1. Ability to Generate and Store all random passwords using one way encryption</td>
<td>✓</td>
</tr>
<tr>
<td>2. Ability to adjusting random password parameters</td>
<td>✓</td>
</tr>
<tr>
<td>3. Password should be truly random with less CPU and Memory overheads</td>
<td>✓</td>
</tr>
<tr>
<td>4. Ability to Limit Login Attempts</td>
<td>✓</td>
</tr>
<tr>
<td>5. Limit Login Period to a few Seconds</td>
<td>✓</td>
</tr>
<tr>
<td>6. Attach Authentication with Randomized, Encrypted Unique Session Token and not Username</td>
<td>✓</td>
</tr>
<tr>
<td>(avoid backend session expiry extension by unethical system administrators)</td>
<td></td>
</tr>
<tr>
<td><strong>Secure Mobile Payment System Integration</strong></td>
<td></td>
</tr>
<tr>
<td>1. Implement Online Transactions Input Channel for Mobile Payment Notifications</td>
<td>✓</td>
</tr>
<tr>
<td>2. Create Access Control List for Source IPN IP</td>
<td>✓</td>
</tr>
<tr>
<td>3. Authenticate IPN Source (validate username and password)</td>
<td>✓</td>
</tr>
<tr>
<td>4. Log sessions (successful and failed authentications)</td>
<td>✓</td>
</tr>
<tr>
<td>5. Send notification (e-mail and SMS) to user on progress status (successful or failed remittances)</td>
<td>✓</td>
</tr>
</tbody>
</table>
4.3 Sample Testing Screenshots

**Fig. 11 - Online Transactions Dashboard**

**Fig. 12 - Sample Receipt Generated by the System**
Fig. 13 - M-Pesa Input Plug-in Transaction Logs

Fig. 14 - Sample Random Password Notification via E-mail
Fig. 15 Random Password Configuration Interface
### 4.4 Sample Mobile Payment Integration Performance Report

#### 4.4.1 Best Performance Tabulation

<table>
<thead>
<tr>
<th>Reference ID</th>
<th>M-Pesa Transaction Time</th>
<th>Transaction Interfacing Time</th>
<th>Interfacing Time Delay (hh:mm:ss)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CU26KC011</td>
<td>3/11/2012 8:38:00</td>
<td>3/11/2012 8:38:43</td>
<td>0:00:43</td>
</tr>
<tr>
<td>DE61UJ310</td>
<td>25/2/2013 8:14:00</td>
<td>25/2/2013 8:14:41</td>
<td>0:00:41</td>
</tr>
<tr>
<td>DR50BL189</td>
<td>8/7/2013 10:04:00</td>
<td>8/7/2013 10:04:40</td>
<td>0:00:40</td>
</tr>
<tr>
<td>DS21SP795</td>
<td>15/7/2013 10:22:00</td>
<td>15/7/2013 10:22:45</td>
<td>0:00:45</td>
</tr>
<tr>
<td>DU32EL919</td>
<td>5/8/2013 8:53:00</td>
<td>5/8/2013 8:53:45</td>
<td>0:00:45</td>
</tr>
<tr>
<td>DV32DC372</td>
<td>14/8/2013 7:57:00</td>
<td>14/8/2013 7:57:43</td>
<td>0:00:43</td>
</tr>
<tr>
<td>DX16CW538</td>
<td>31/8/2013 8:04:00</td>
<td>31/8/2013 8:04:37</td>
<td>0:00:37</td>
</tr>
<tr>
<td>DX17QO820</td>
<td>31/8/2013 10:24:00</td>
<td>31/8/2013 10:24:28</td>
<td>0:00:28</td>
</tr>
<tr>
<td>DX38XL338</td>
<td>2/9/2013 10:31:00</td>
<td>2/9/2013 10:31:44</td>
<td>0:00:44</td>
</tr>
<tr>
<td>DX39JY452</td>
<td>2/9/2013 11:05:00</td>
<td>2/9/2013 11:05:38</td>
<td>0:00:38</td>
</tr>
<tr>
<td>DX50WZ387</td>
<td>3/9/2013 10:17:00</td>
<td>3/9/2013 10:17:34</td>
<td>0:00:34</td>
</tr>
<tr>
<td>DX64PB808</td>
<td>4/9/2013 12:15:00</td>
<td>4/9/2013 12:15:48</td>
<td>0:00:48</td>
</tr>
<tr>
<td>DX73AG578</td>
<td>5/9/2013 7:33:00</td>
<td>5/9/2013 7:33:49</td>
<td>0:00:49</td>
</tr>
<tr>
<td>DY08HU346</td>
<td>8/9/2013 8:47:00</td>
<td>8/9/2013 8:47:48</td>
<td>0:00:48</td>
</tr>
<tr>
<td>DY17JC367</td>
<td>9/9/2013 5:58:00</td>
<td>9/9/2013 5:58:37</td>
<td>0:00:37</td>
</tr>
<tr>
<td>EB04YU517</td>
<td>5/10/2013 10:43:00</td>
<td>5/10/2013 10:43:30</td>
<td>0:00:30</td>
</tr>
<tr>
<td>EB28VX989</td>
<td>7/10/2013 11:50:00</td>
<td>7/10/2013 11:50:36</td>
<td>0:00:36</td>
</tr>
<tr>
<td>EB38RO730</td>
<td>8/10/2013 8:58:00</td>
<td>8/10/2013 8:58:35</td>
<td>0:00:35</td>
</tr>
<tr>
<td>EB53VK284</td>
<td>9/10/2013 12:24:00</td>
<td>9/10/2013 12:24:39</td>
<td>0:00:39</td>
</tr>
<tr>
<td>EC42NU400</td>
<td>17/10/2013 7:16:00</td>
<td>17/10/2013 7:16:49</td>
<td>0:00:49</td>
</tr>
<tr>
<td>EE44FH138</td>
<td>4/11/2013 9:33:00</td>
<td>4/11/2013 9:33:31</td>
<td>0:00:31</td>
</tr>
<tr>
<td>EE55NM647</td>
<td>5/11/2013 7:51:00</td>
<td>5/11/2013 7:51:48</td>
<td>0:00:48</td>
</tr>
<tr>
<td>EE57RD010</td>
<td>5/11/2013 10:44:00</td>
<td>5/11/2013 10:44:37</td>
<td>0:00:37</td>
</tr>
<tr>
<td>EL74AL106</td>
<td>6/1/2014 7:31:00</td>
<td>6/1/2014 7:31:42</td>
<td>0:00:42</td>
</tr>
<tr>
<td>EM35VY395</td>
<td>11/1/2014 10:32:00</td>
<td>11/1/2014 10:32:28</td>
<td>0:00:28</td>
</tr>
</tbody>
</table>
4.4.2 Best Performance Kernel Density Plot

![Graph](image)

**Fig. 16 – Mobile Payment Integration Performance Best Performance Graph**

The graph illustrates a negatively (left) skewed distribution. The optimum interfacing time (delay) for the transactions is 39 seconds.
## 4.4.3 Worst Performance Tabulation

<table>
<thead>
<tr>
<th>Reference ID</th>
<th>M-Pesa Transaction Time</th>
<th>Transaction Interfacing Time</th>
<th>Interfacing Time Delay (hh:mm:ss)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DD93ZA504</td>
<td>17/2/2013 7:43:00</td>
<td>17/2/2013 20:30:31</td>
<td>12:47:31</td>
</tr>
<tr>
<td>DE01NQ481</td>
<td>18/2/2013 4:55:00</td>
<td>18/2/2013 18:40:11</td>
<td>13:45:11</td>
</tr>
<tr>
<td>DE01RE433</td>
<td>18/2/2013 5:06:00</td>
<td>18/2/2013 18:42:41</td>
<td>13:36:41</td>
</tr>
<tr>
<td>DE02TH039</td>
<td>18/2/2013 6:31:00</td>
<td>18/2/2013 18:56:23</td>
<td>12:25:23</td>
</tr>
<tr>
<td>DO33DB154</td>
<td>7/6/2013 3:01:00</td>
<td>7/6/2013 16:58:30</td>
<td>13:57:30</td>
</tr>
<tr>
<td>DO33NX447</td>
<td>7/6/2013 3:35:00</td>
<td>7/6/2013 17:11:48</td>
<td>13:36:48</td>
</tr>
<tr>
<td>DO34LF273</td>
<td>7/6/2013 4:45:00</td>
<td>7/6/2013 18:11:05</td>
<td>13:26:05</td>
</tr>
<tr>
<td>DO34TK739</td>
<td>7/6/2013 5:07:00</td>
<td>7/6/2013 18:30:13</td>
<td>13:23:13</td>
</tr>
<tr>
<td>DO87FU998</td>
<td>12/6/2013 5:27:00</td>
<td>12/6/2013 17:59:40</td>
<td>12:32:40</td>
</tr>
<tr>
<td>DS35QP968</td>
<td>16/7/2013 3:00:00</td>
<td>16/7/2013 19:05:59</td>
<td>16:05:59</td>
</tr>
<tr>
<td>DT49MQ102</td>
<td>28/7/2013 7:13:00</td>
<td>29/7/2013 9:24:22</td>
<td>26:11:22</td>
</tr>
<tr>
<td>DT56GF377</td>
<td>28/7/2013 7:28:00</td>
<td>29/7/2013 9:24:18</td>
<td>25:56:18</td>
</tr>
<tr>
<td>DU06VC703</td>
<td>2/8/2013 5:58:00</td>
<td>2/8/2013 18:22:01</td>
<td>12:24:01</td>
</tr>
</tbody>
</table>
4.4.4 Worst Performance Kernel Density Plot

![Graph of Mobile Payment Integration Performance Worst Performance](image)

**Fig. 17 – Mobile Payment Integration Performance Worst Performance Graph**

The graph illustrates a positively (right) skewed distribution. The worst case interfacing time (delay) for the transactions is 12.5 hours.
4.4.5 Mobile Payment Integration Report Interpretation

The overall performance of the Mobile Money Payment System integration is dependent on M-Pesa notification system, communication channels (internet) and transaction processing engine. Despite best or worst case throughput, the online transaction engine maintains below 60 seconds lead time. However, M-Pesa notifications fluctuate over time. Since Transaction Engine’s parameters (bandwidth, memory, system uptime and processor) have been constant, we may conclude that the bottleneck is likely to be originating from the M-Pesa notification side.

The notification delays (which are beyond our control) may be attributed to the following:-

- M-Pesa (weekly) system upgrade
- Internet connectivity outage
- Poor bandwidth
### 4.5 Sample Random Password SMS Performance Report

#### 4.5.1 Best Performance Tabulation

<table>
<thead>
<tr>
<th>BulkSMSID</th>
<th>Attempt</th>
<th>Registration Date</th>
<th>Delivery Date</th>
<th>Delivery Time Delay (hh:mm:ss)</th>
</tr>
</thead>
<tbody>
<tr>
<td>55579</td>
<td>1</td>
<td>5/12/2013 12:58:00</td>
<td>5/12/2013 12:58:10</td>
<td>0:00:10</td>
</tr>
<tr>
<td>55583</td>
<td>1</td>
<td>8/12/2013 10:01:00</td>
<td>8/12/2013 10:01:07</td>
<td>0:00:07</td>
</tr>
<tr>
<td>57394</td>
<td>1</td>
<td>11/12/2013 14:24:00</td>
<td>11/12/2013 14:24:09</td>
<td>0:00:09</td>
</tr>
<tr>
<td>58032</td>
<td>1</td>
<td>5/1/2014 11:51:00</td>
<td>5/1/2014 11:51:10</td>
<td>0:00:10</td>
</tr>
<tr>
<td>58040</td>
<td>1</td>
<td>5/1/2014 17:51:00</td>
<td>5/1/2014 17:51:10</td>
<td>0:00:10</td>
</tr>
<tr>
<td>58052</td>
<td>1</td>
<td>6/1/2014 7:32:00</td>
<td>6/1/2014 7:32:10</td>
<td>0:00:10</td>
</tr>
<tr>
<td>58090</td>
<td>1</td>
<td>6/1/2014 18:16:00</td>
<td>6/1/2014 18:16:09</td>
<td>0:00:09</td>
</tr>
<tr>
<td>58150</td>
<td>1</td>
<td>7/1/2014 15:50:00</td>
<td>7/1/2014 15:50:09</td>
<td>0:00:09</td>
</tr>
<tr>
<td>58230</td>
<td>1</td>
<td>8/1/2014 9:38:00</td>
<td>8/1/2014 9:38:10</td>
<td>0:00:10</td>
</tr>
<tr>
<td>58250</td>
<td>1</td>
<td>8/1/2014 14:16:00</td>
<td>8/1/2014 14:16:09</td>
<td>0:00:09</td>
</tr>
<tr>
<td>58259</td>
<td>1</td>
<td>8/1/2014 18:21:00</td>
<td>8/1/2014 18:21:09</td>
<td>0:00:09</td>
</tr>
<tr>
<td>58271</td>
<td>1</td>
<td>9/1/2014 6:55:00</td>
<td>9/1/2014 6:55:10</td>
<td>0:00:10</td>
</tr>
<tr>
<td>58316</td>
<td>1</td>
<td>11/1/2014 12:14:00</td>
<td>11/1/2014 12:14:10</td>
<td>0:00:10</td>
</tr>
<tr>
<td>58371</td>
<td>1</td>
<td>15/1/2014 19:20:00</td>
<td>15/1/2014 19:20:10</td>
<td>0:00:10</td>
</tr>
<tr>
<td>58461</td>
<td>1</td>
<td>21/1/2014 6:56:00</td>
<td>21/1/2014 6:56:09</td>
<td>0:00:09</td>
</tr>
<tr>
<td>58965</td>
<td>1</td>
<td>25/1/2014 16:03:00</td>
<td>25/1/2014 16:03:10</td>
<td>0:00:10</td>
</tr>
<tr>
<td>58968</td>
<td>1</td>
<td>25/1/2014 17:51:00</td>
<td>25/1/2014 17:51:10</td>
<td>0:00:10</td>
</tr>
<tr>
<td>59581</td>
<td>1</td>
<td>29/1/2014 20:53:00</td>
<td>29/1/2014 20:53:10</td>
<td>0:00:10</td>
</tr>
<tr>
<td>59591</td>
<td>1</td>
<td>30/1/2014 10:21:00</td>
<td>30/1/2014 10:21:10</td>
<td>0:00:10</td>
</tr>
<tr>
<td>59616</td>
<td>1</td>
<td>31/1/2014 2:23:00</td>
<td>31/1/2014 2:23:10</td>
<td>0:00:10</td>
</tr>
<tr>
<td>59763</td>
<td>1</td>
<td>31/1/2014 13:31:00</td>
<td>31/1/2014 13:31:08</td>
<td>0:00:08</td>
</tr>
<tr>
<td>59777</td>
<td>1</td>
<td>31/1/2014 18:16:00</td>
<td>31/1/2014 18:16:09</td>
<td>0:00:09</td>
</tr>
<tr>
<td>60186</td>
<td>1</td>
<td>5/2/2014 12:01:00</td>
<td>5/2/2014 12:01:09</td>
<td>0:00:09</td>
</tr>
<tr>
<td>61316</td>
<td>1</td>
<td>17/2/2014 23:22:00</td>
<td>17/2/2014 23:22:09</td>
<td>0:00:09</td>
</tr>
<tr>
<td>65126</td>
<td>1</td>
<td>19/3/2014 14:00:00</td>
<td>19/3/2014 14:00:08</td>
<td>0:00:08</td>
</tr>
</tbody>
</table>
4.5.2 Best Performance Kernel Density Plot

The graph illustrates a negatively (left) skewed distribution. The optimum delivery time (delay) for the Random Password SMS is 10 seconds.
### 4.5.3 Worst Performance Tabulation

<table>
<thead>
<tr>
<th>BulkSMSID</th>
<th>Attempt</th>
<th>Registration Date</th>
<th>Delivery Date</th>
<th>Delivery Time Delay (hh:mm:ss)</th>
</tr>
</thead>
<tbody>
<tr>
<td>55554</td>
<td>1</td>
<td>1/12/2013 16:33:00</td>
<td>3/12/2013 10:14:16</td>
<td>41:41:16</td>
</tr>
<tr>
<td>56080</td>
<td>1</td>
<td>10/12/2013 0:34:00</td>
<td>11/12/2013 8:44:39</td>
<td>32:00:39</td>
</tr>
<tr>
<td>57429</td>
<td>1</td>
<td>11/12/2013 14:24:00</td>
<td>12/12/2013 22:24:39</td>
<td>32:00:39</td>
</tr>
<tr>
<td>57487</td>
<td>1</td>
<td>11/12/2013 14:24:00</td>
<td>12/12/2013 23:05:52</td>
<td>32:41:52</td>
</tr>
<tr>
<td>57528</td>
<td>1</td>
<td>11/12/2013 14:48:00</td>
<td>12/12/2013 22:48:55</td>
<td>32:00:55</td>
</tr>
<tr>
<td>57586</td>
<td>1</td>
<td>11/12/2013 14:48:00</td>
<td>12/12/2013 23:05:43</td>
<td>32:17:43</td>
</tr>
<tr>
<td>57697</td>
<td>1</td>
<td>18/12/2013 15:59:00</td>
<td>20/12/2013 0:00:05</td>
<td>32:01:05</td>
</tr>
<tr>
<td>58669</td>
<td>1</td>
<td>22/1/2014 9:42:00</td>
<td>23/1/2014 17:43:19</td>
<td>32:01:19</td>
</tr>
<tr>
<td>59077</td>
<td>1</td>
<td>29/1/2014 8:58:00</td>
<td>30/1/2014 16:59:47</td>
<td>32:01:47</td>
</tr>
<tr>
<td>61303</td>
<td>1</td>
<td>17/2/2014 21:24:00</td>
<td>19/2/2014 5:58:06</td>
<td>32:34:06</td>
</tr>
<tr>
<td>63555</td>
<td>1</td>
<td>19/3/2014 13:28:00</td>
<td>20/3/2014 21:30:00</td>
<td>32:02:00</td>
</tr>
<tr>
<td>64352</td>
<td>1</td>
<td>19/3/2014 13:33:00</td>
<td>20/3/2014 21:38:02</td>
<td>32:05:02</td>
</tr>
<tr>
<td>65366</td>
<td>1</td>
<td>2/4/2014 9:04:00</td>
<td>3/4/2014 18:06:05</td>
<td>33:02:05</td>
</tr>
</tbody>
</table>
4.5.4 Worst Performance Kernel Density Plot

The graph illustrates a positively (right) skewed distribution. The worst case delivery time (delay) for the Random Password SMS is 32 hours.
4.5.5 Random Password SMS Report Interpretation

Current queuing time for SMS notification is approximately 5 seconds. It is evident that the notification can be delivered and status confirmed within 10 seconds; 5 seconds after the message has been queued. Routing is based on HTTP traffic and is dependent on connectivity between the 3rd Party Bulk SMS Service Provider and the subscriber’s Mobile Service Provider.

Worse case report indicates that the delivered SMS is less than 48 hours old. This justifies the fact that standard SMS expiry time is 2 days. Otherwise the SMS status would be failed or rejected.

Once again, the SMS notification delays (which are beyond our control) may be attributed to the following:-

- Lack of coverage
- Lack of mobile device power
- SMS Barring
- Lack of internet connectivity
Chapter 5: Summary and Recommendations

5.1 Current Limitations

- M-Pesa IPN currently provides One-Way Input Channel via HTTP GET with a few attempts
- The system uses single-threaded channel without priority queuing to send SMS. In case of long queues, the password might be delayed and consequently received after session has expired

5.2 Other and Future Enhancement

- In case of service outage (Internet and Server downtime), the client needs to pull notifications from the IPN server hence the need for a 2 way integration channel. This is also necessary for counter checking received transactions
- Secondly, to avoid sending expired Random Passwords to users, priority queuing logic (in conjunction with First In First Out) needs to be added so as to prioritize relay of Random Password SMS

5.3 Best Configuration Settings

In order to highly secure Random Password Authentication, one needs to input strong configuration settings that minimize chances of brute force attack. The following are sample settings to enhance security:

- Increase the Random Password Length to a higher figure
- Reduce Authentication Attempts
- Set notification channel to SMS only
- Reduce Password Expiry Duration to a few seconds
- Enable Priority Queuing Logic for sending Random Password SMS notifications

5.4 Conclusion

This Project is a roadmap for the creation of secure, local e-commerce solutions using Mobile Payment Systems.
Bibliography


Diane, Cabell. FAQs.org (2001). E-Commerce: An Introduction


http://en.wikipedia.org/wiki/One-time_password

http://en.wikipedia.org/wiki/Software_development_methodology

http://en.wikipedia.org/wiki/Software_prototyping