Enhanced Security Model For Mobile Banking Systems In Tanzania

Baraka W. Nyamtiga, Anael Sam, Loserian S. Laizer

School of Computational and Communication Science and Engineering, The Nelson Mandela African Institution of Science and Technology (NM-AIST), Arusha, Tanzania.
Email: nyamtigab@nm-aist.ac.tz

ABSTRACT: In mobile banking schemes; financial services are availed and banking services are provided using mobile devices. GSM services are greatly utilized for data transmission by the technologies used in conducting mobile transactions. In their operations; these technologies send data in plaintext. Financial service providers tend to rely on the security services provided by the GSM which has been proved to be susceptible to cryptanalytic attacks. The used algorithms for crypto mechanisms are flawed leaving data carried through the network vulnerable upon interception. Operators need to take precaution by enforcing some protective measures on the information to be transmitted. This paper describes an SMS based model designed with security features to enhance data protection across mobile networks. Features for data encryption, integrity, secure entry of security details on the phone, and improved security policies in the application server are incorporated. We address issues of data confidentiality, user authentication and message integrity in order to provide end-to-end security of data carried on GSM networks.

Keywords: Cryptography; Data Integrity; GSM; Message Confidentiality; Mobile Banking; Security; User Authentication

1 INTRODUCTION

Mobile services are highly adopted in developing countries because of the rapid growth in mobile networks. It is estimated by *The International Telecommunication Union* (ITU) that at the end of the year 2012 there were 6.8 billion global mobile subscriptions and it is predicted by *Portion Research* in its, "Mobile Factbook 2013" that by the end of 2016 the global subscribers will reach 8.5 billion [1]. According to the *Tanzania Communications Regulatory Authority* (TCRA); total subscriptions in Tanzania by march 2013 were around 27.6 million [2] as compared to 15 million subscribers in 2009 [3]. This shows that mobile technologies are rapidly being adopted (both locally and globally) and along with it; a significant growth in mobile services evolution. The fast increasing number of subscribers open up new business ventures and gives financial institutions some additional channels for them to deliver their services. Wikipedia [4] defines mobile banking as the provision and availment of banking- and financial services with the help of mobile telecommunication devices. These schemes are highly promoted recently in Tanzania and they come with varieties of services; withdrawal/deposit of money into accounts, money transfers, payment of bills, etc. The most famous of these are *mobile money transfer services* offered by telecommunications operators and *mobile banking systems* offered by banking institutions. Some examples of these systems and services include *Tigopesa, M-Pesa, Simbanking, and NMB Mobile* offered by *Tigo Tanzania Ltd, Vodacom Tanzania, CRDB Bank,* and *National Microfinance Bank* respectively. These mobile transactions schemes offer customers convenience to access their financial resources timely and with much ease while the service providers generate some extra revenue. Over the past 5 years, Tanzania has seen a significant growth in these schemes that is still increasing. However, these systems face a number of challenges related to security ranging from risk behaviors done by users ignorantly, system availability, and some cases of fraud in which money vanish from customers' accounts without their knowledge. This calls a need for careful measures being taken by respective parties involved in order to ensure the safety of customer information and ensure that transactions are conducted securely. *WAP* (Wireless Application Protocol) best described as mobile internet, *SMS* (Secure Messaging Service), *USSD* (Unstructured Supplementary Services Data), and *IVR* (Interactive Voice Response) are among the technologies that are used in conducting mobile transactions. These mobile services are widely deployed over GSM networks whose initial design was meant for voice traffic. A range of GSM generations are now in use which include 2G (Second Generation), GPRS (General Packet Radio Services), EDGE (Enhanced Data Rates for GSM Evolution), UMTS (Universal Mobile Telecommunications System), 3G (Third Generation), and LTE (Long Term Evolution). The higher generations cover the city areas, major towns and some strategic remote areas. But they are all backward compatible with the 2G systems whose coverage is wider. Ultimately, for mobile transactions to reach most customers the majority of which are in rural areas 2G GSM systems are being utilized in Tanzania and most of the developing countries. The use of GSM for message exchange between the customer and the server puts the transactions at a risk of interception because the initial designs of GSM were meant for subscribers to exchange non-sensitive messages. Security implementations for mutual authentication, end-to-end security, data confidentiality, and non-repudiation were omitted as for message services [5]. As such sensitive financial details are carried across GSM; issues of security arise as the information becomes exposed to security attacks. The main security issues in GSM are related to encryption and authentication algorithms. The commonly used algorithm for encryption, A5 has been successfully reverse engineered [6], [7], [8]. Also several flaws have been spotted in the A3/A8 authentication algorithm which makes it possible to break it and obtain a root key *Ki* and the mobile subscriber identity, *IMSI* (International Mobile Subscriber Identity) [9], [10]. Obtaining *Ki* and *IMSI* makes it possible to clone the SIM in a couple of hours using freely available tools. These vulnerabilities make the technologies that utilize GSM services susceptible to attacks if data is not well protected. Most of the mobile transactions services in developing countries make use of SMS and USSD technologies. In Tanzania, the service providers are mostly using USSD in conducting the transactions. The report given by United Nations Conference on Trade and Development (UNCTAD) regarding, "*Mobile Money for Business Development in the East African*

Community" assert that data sent via SMS or USSD is vulnerable for interception if it is not well protected [11].

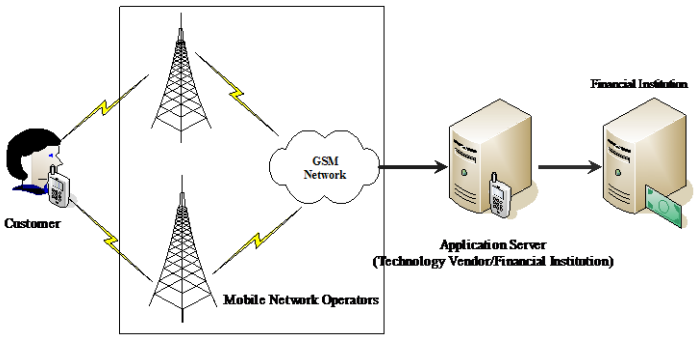

Fig. 1: The different entities involved in mobile transactions

To perform transactions using a mobile device; a customer initiates the request that goes through the *mobile network operator (*MNO*)* and terminates at the *server application* that can be administered by a *technology vendor* or independently by a *financial institution*. The parties involved in mobile transactions are diagrammatically illustrated in Fig. 1 above. Between the customer and application server, data is transmitted across GSM network. Because of the security flaws that exist in GSM, the financial institutions are urged to secure the messages with adequate measures prior to sending them [12]. Securing the messages before are sent ensures the end-to-end security, should data be intercepted in transit; the intercepted messages are just useless to the interceptor. We discuss in this paper a number of security measures to be taken in order to enhance the security controls of mobile banking systems.

2 Literature Review

In this section, research publications related to security in mobile banking systems and GSM are reviewed and analyzed. The focus is on issues related to provision of end-to-end security, mutual authentication, message integrity and effective server security management policies. Hash functions, message authentication codes and digital signatures mechanisms are used by Van der Merwe [13] in securing financial transactions over distrusted networks. He does an evaluation of GSM security and assessment of potential attacks to understand risks associated with performing mobile commerce transactions using GSM. He discusses the mistakes done by the GSM Consortium developing their security technologies in secret, rather than in public domain for peer review. This was a major reason these systems came under attack eventually. He puts an emphasis on protecting transmissions that involve financial transactions because attackers could generate huge benefits by altering or fabricating the messages transmitted. In his solution, he makes use of standard SMS with the assistance of SIM application Toolkit (STK). He addresses message integrity issues by deriving a *Message Authentication Code* (MAC) from a message by feeding it through a *Triple DES* (Data Encryption Standard) algorithm which is sent along with the original message. He makes use the SIM Card encryption to ensure confidentiality by encrypting the PIN and argues that taking these measures makes transacting over wireless networks more confident. It is described by Chong [14] that despite the built-in security mechanisms in GSM to prevent external attackers reading contents of messages they intercept; there are weaknesses at the SMS Center (SMSC). At the SMSC; messages waiting to be delivered are stored in plaintext format, and even after the messages are delivered; records of all messages are kept by the cellular operator. Using plaintext messages is not sufficiently secure because some alleged employee at the operator can access these messages at the SMSC. Chong designs a secure messaging protocol using SMS via GSM and integrates this protocol with mobile banking systems. The key management for the cryptographic mechanisms remains to be a challenge that he chose to neglect and leave some parts of the message unencrypted as they are being sent. The higher security capabilities provided with packet switched networks are described by Chikomo [15] as a contributing factor to steady increase in popularity of mobile banking over GPRS. Chikomo raises questions of the GPRS security implementations in place to cater for secure communications and the security protocols in place for confidentiality and the integrity of information being communicated. He identifies the additional security layers implemented in GPRS using the overlying protocols to enforce the security of data. These protocols are such as *Wireless Application Protocol* (WAP) and *I-Mode* (a Japanese version of WAP). WAP provides security of communications using WAP Transport Layer Security (WTLS) while I-Mode uses Secure Sockets Layer (SSL) for its security. Ultimately, he discusses the resolutions for improving WAP banking implementations using a new security protocol that he calls *Secure GPRS Protocol* (SGP). The protocol ensures addresses confidentiality by detecting any modifications and ensures that client and server authenticate each before sharing confidential information. Another SMS solution is provided by Emmanuel and Jacobs [5] to enable customers transact securely using encrypted and signed messages across the GSM. He uses *Advanced Encryption Standard* (AES) symmetric encryption algorithm to encrypt messages before transmission. They propose further research for optimized asymmetric cryptosystems that will be suitable for low memory capacity mobile terminals. Chong [16] in his master's thesis investigates the usable authentication mechanisms for mobile banking. He describes some usability concerns regarding the use of PIN method for authentication, memorability issues being one of them. Chong designs two techniques for mobile authentication; *combinational graphical passwords* and *gesture passwords*. He implements the prototypes for these techniques and evaluated them with users along with PIN authenticator. Based on user experiences and password retention, the evaluations revealed that users are more proficient and would prefer to use PINs for authentication. Despite gesture and graphical passwords seeming to resolve the memorability issues; he concludes that PIN remains to be the more preferred mechanism that is trusted and convenient to users than the others. Another authentication mechanism is proposed by Bilal and Sankar [17] that is based on biometric mechanisms. Bilal uses biometric fingerprint scans in his design to authenticate users in the system which appeals to be more effective in confirming the user's identity. However, its practical implementation requires a separate fingerprint device being connected to the phone or the phone manufacturers incorporating a fingerprint scanning functionality in the handsets. The practical application for this approach in developing countries is not feasible. The authors in [18] explore different authentication

schemes and propose a new scheme for user authentication in banking systems that offers better secrecy of PINs. The scheme that is proposed by them is based on a numeric 4-digit PIN and a printed codebook containing a list of 6-digit nonces. They improve this scheme and use a codebook with nonces of *10-digit* number each. A new approach to secure mobile transactions using *Public Key Infrastructure* (PKI) is provided by Narendiran [19]. PKI is an asymmetric encryption scheme in which each device has a private key and a public key that is shared among all devices. When data is encrypted using the public key; only the private key can decrypt such data. Key management is identified to be a big challenge in the large volume of users and Narendiran proposes a distribution of the client certificate using some secure communication through an online banking server after the user has registered for the service. Using appropriate interface connectivity, a direct communication is established by the application between the mobile phone and the bank database. The communication encrypts data from the mobile phone and decrypted at the server, and the same happens in data retrieval from the server making eavesdropping impossible. In their review paper for security issues in GSM and its mitigation, Kaur, Kaur et al. [7] describe the various attacks on GSM networks which include cryptographic attacks, denial of service, brute force, and replay attacks. Among the mitigations measures he suggests include the use of a new stronger encryption to replace A5/1 & A5/2 and a more cryptographically secure algorithm to replace A3 which will imply a distribution of new SIM Cards to all subscribers. In his academic PhD dissertation titled *Secure Mobile Service-Oriented Architecture* Zhang [20] finds security to be among the most important issues hindering wider deployment and acceptance of mobile systems, especially when dealing with highly sensitive details; like financial transactions. His research focuses on the security architecture for mobile environments and he addresses issues of adaptable and secure infrastructure for mobile services and applications as well as resource limitations of mobile devices and networks. In his design for a secure mobile service-oriented architecture, Zhang integrates several financial services in which secure messages are exchanged between the different infrastructural components. Zhang's generic infrastructure can be adopted by various types of mobile transactions including *m-Banking*, *m-Ticketing*, *m-Commerce*, *m-Loans,* etc. The services provided in his architecture include *light-weight* security services suitable for mobile devices, networks and applications, *mobile PKI* based on mobile trusted authorities and he finally designs and develops several secure mobile applications.

3 GSM Overview and its Security Mechanisms

3.1 GSM Overview

The Global System for Mobile Communications (GSM) began in 1982 as Groupe Spéciale Mobile formed for the purpose of investigating and developing a public land mobile system. This was motivated by the rapid growth of analogue cellular systems in Europe in 1980's with each country developing different systems; incompatible with all others in equipment and operation [13]. Other concerns involved critical security issues related to cell phone cloning in which case fake base stations were utilized [21]. GSM foundation marked the beginning of *second-generation* (2G) technology. GSM's general architecture is composed of several functional entities and can be divided into three broad parts: The *Mobile Station,* the *Base Station Subsystem* and the *Network Subsystem.* The Mobile Station (MS) is carried by a subscriber and it has a special smartcard called *Subscriber Identity Module* (SIM) which allows the user to access the subscribed services. The SIM contains an *International Mobile Subscriber ID,* IMSI for identification of the mobile on GSM network [21]. Communications are initiated by the MS and signals are transmitted to the *Base Transceiver Station* (BTS). The Base Station Subsystem (BSS) is used to control the radio link with the MS and it consists of the BTS and the *Base Station Controller* (BSC). Radio signals are translated by the BTS into digital format and the digital signals are then transferred to the BSC. The BSC manages radio resources for one or more BTSs and it forwards the signals it receives to the *Mobile Switching Center* (MSC). In the Network Subsystem; calls are switched between the mobile and the fixed network users. The MSC, *SMS Center* (SMSC)*, International Switching Center* (ISC)*, Home Location Register* (HLR)*, Visitor Location Register* (VLR)*, Equipment Identity Register* (EIR) and the *Authentication Center* (AuC) are all located in this part. Issues of registration, authentication, location updating, and call routing to roaming subscribers are handled by the MSC. The HLR keeps track of the mobile's location and contain information for registered subscribers on a network [13]. Calls for subscribed services for each mobile within an area are controlled by the VLR which contains selective administration information from the HLR. Records for all valid mobiles in the network, each of which is identified by its *International Mobile Equipment Identity* (IMEI) are kept in the *Equipment Identity Register* (EIR) database. A copy of the root key found in each SIM card that is used for cryptographic operations over the radio channel is stored in a protected database called *Authentication Center* (AuC). When signals are received by the BSC from the BTS, they are forwarded to the MSC which interrogates its HLR and VLR databases for location information about the destination [5]. If the signal originates or terminates at the fixed line network it will be routed to the *Gateway MSC* (GMSC). If the signal is an SMS message then it will be stored in the SMSC and wait to be delivered to destination. A copy of the SMS is kept in the SMSC's persistent database even after being delivered and the signal is routed to another country via the ISC if it needs to be redirected internationally. Maintenance Operations are controlled by the *Operation and Management Center* (OMC). Equipment verification and user authentication are respectively facilitated by the EIR and the AUC databases.

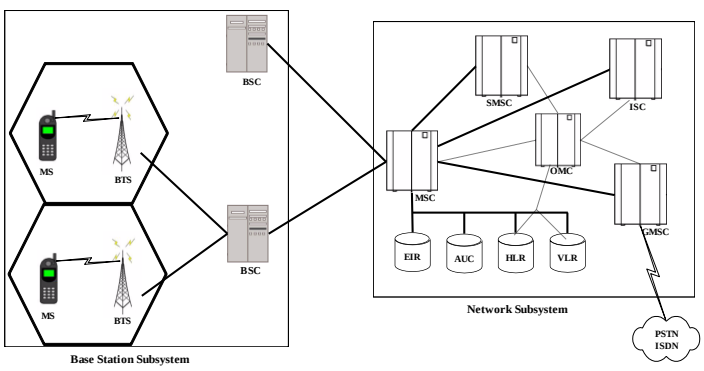

Fig. 2: GSM Network Architecture

3.2 Overview of Security in GSM

[21] identifies two primary security goals set forth by GSM designers which are; to make GSM as secure as ordinary PSTN[1] telephones and prevent phone cloning. These goals are achieved by addressing three security issues; *anonymity, authentication, and confidentiality.* Anonymity is designed to ensure customer confidence whereas authentication is vital to phone companies for proper billing and confidentiality of communications over the air interface is important to both the phone companies and the customers [21]. Customer *anonymity* provides confidentiality for user identity, his location as well signaling information by ensuring that the IMSI or any other information that can be used to derive it is not transmitted in plaintext on the radio path. A *Temporary Mobile Subscriber Identity* (TMSI) is used instead for subscriber identification with an exception for the initial contact attempt during the setup phase. The VLR is used for TMSI to IMSI mapping and TMSI is only valid in a single temporary session within a specific location area; it is securely updated after every successful GSM access. Using the subscriber's IMSI; the VLR database is interrogated for the TMSI corresponding to the user which the operator uses to compare with user's TMSI for authentication. The next TMSI will be sent to the MS for next authentication at the end of every successful authentication and it is usually encrypted prior to being sent. It is critically necessary for the operator to be able to authenticate the user on the network to ensure that they get paid for their services. According to Stamp, lack of mutual authentication is one of the significant security oversights of GSM. Only the subscriber is authenticated to the base station and the BTS is not authenticated. A simple challenge-response mechanism is utilized in GSM for authentication using the subscriber's authentication key, K_i and the IMSI [10]. The Key *Ki* and IMSI are stored by the operator both in the SIM card and the Authentication Center (AUC). They are used as inputs in the A3 algorithm which is used for authentication. During a setup process, the subscriber's IMSI is received by the BTS and passed over to the home network which generates a random challenge, RAND, and computes a 32-bit output called *signal response*, $SRES_{NW}$ = A3(RAND, K_i). The pair (RAND, $SRES_{NW}$) is then sent from the network to the BTS which sends the RAND to the MS. From the received challenge, the mobile also computes another signal response as $SRES_{MS}$ = A3 (RAND, K_i) and sends it back to the BTS which verifies that $SRES_{MS}$ = $SRES_{NW}$. The authentication is successful only if the two SRES values match and otherwise it is a failure. During the authentication process, the key *Ki* never leaves the SIM or the home network which promises a degree of security in the process and Fig. 3 below illustrates the whole process of authentication.

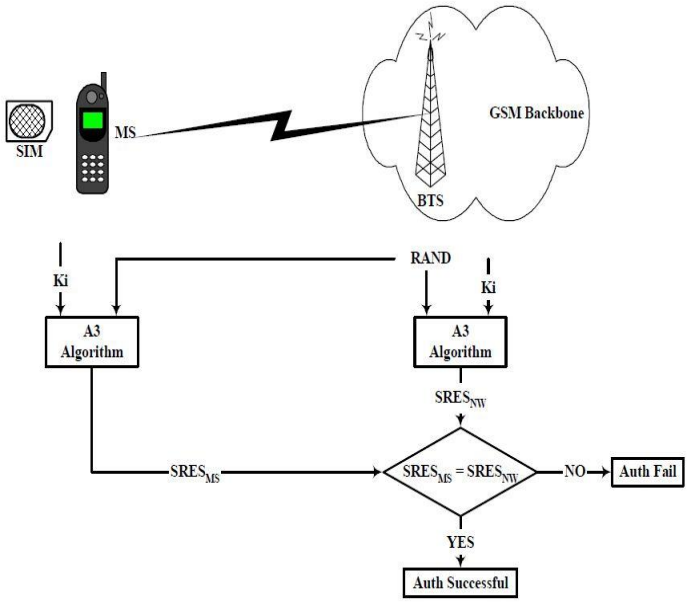

Fig. 3: GSM Authentication Protocol

Confidentiality ensures user privacy for information carried in signaling and traffic channels. It is achieved in GSM by encrypting the transmitted data using a stream cipher called A5. This cipher has multiple versions which include *A5/0* that doesn't utilize any encryption and used in countries with political issues in cryptographic hardware, *A5/1* which is deemed to be strong and used in Europe and North America. Other versions are *A5/2*; a weaker encryption algorithm created for export and *A5/3* which is newer, stronger cipher under UMTS to secure 3G services [22]. The A8 algorithm is used to generate a symmetric key, K_c for data encryption in the A5 cipher. The Key *Kc* is computed by the home network after receiving the IMSI from the BSC as K_c = A8 (RAND, K_i) and it is then sent along with the pair (RAND, $SRES_{NW}$) now constituting a triple (RAND, $SRES_{NW}$, K_c) to the BTS. The BTS performs some authentication checks and if the subscriber is authentic; the MS also computes the key, K_c = A8 (RAND, K_i). The mobile and the BTS now have a shared key to encrypt the communications between them because the BTS already knows *Kc* from the network. The data and the key *Kc* are used as inputs to the A5 ciphering algorithm which symmetrically encrypts the data to produce the ciphertext as the output. Then the ciphertext is converted into radio signals by the MS and sent over the air to the BTS. Fig. 4 below illustrates GSM encryption.

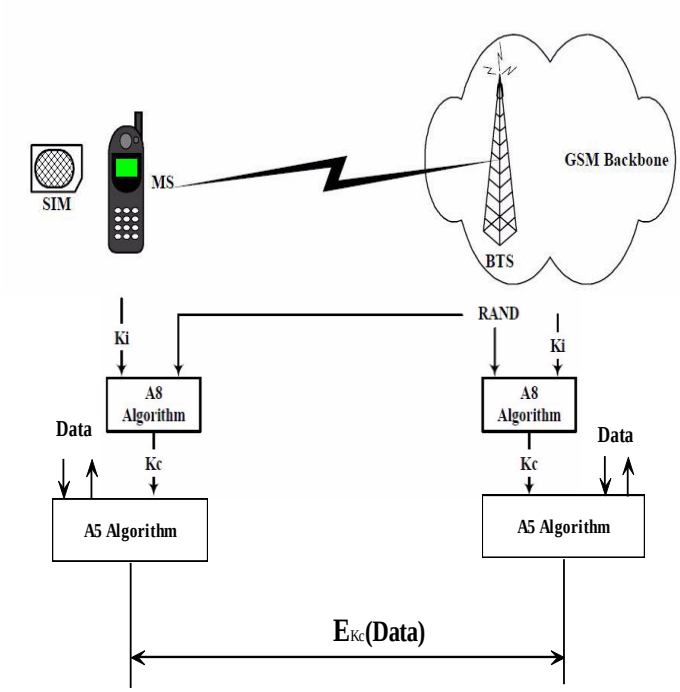

Fig. 4: GSM A5 Encryption

3.3 GSM Security Issues

Besides the improvements in security over the first-generation cellular (1G) systems; GSM security has come under strong criticism as it matures and expands [13]. Its major criticism lies with its development of cryptographic algorithms in secrecy. This made it inevitable for GSM to come under attack for its dependency on the A3, A8, and A5 proprietary algorithms. Security of a system should depend on the secrecy of keys and not of algorithms. Among other flaws as discussed below, these algorithms are believed to be cryptographically weak, and vulnerable to attacks by well-equipped hackers [21]. Some weaknesses in GSM security are described below.

SIM Attacks

The A3 and A8 cryptographic algorithms used in GSM are both implemented using a hash function known as COMP128; this makes them both rendered weak. COMP128's design was developed in secrecy violating Kerckhoffs' Principle[2]. It was eventually reverse engineered at Berkeley in 1998 meaning that an attacker with access to a SIM card can determine the root key *Ki*. The key was successfully obtained by cryptanalysis of COMP128 performed by Briceno, Goldberg, and Wagner allowing the phone to be cloned [23], [24]. An alleged seller, for example, could determine *Ki* using a SIM card reader before selling the phone. The *Ki* and the IMSI can then be written by the attacker onto another SIM card which can be used to execute masquerading attacks.

Encryption Attacks

Because of export restrictions on encryption technology, the stronger A5/1 ciphering algorithm is used only in western Europe and a few others while the weaker A5/2 is used in central and eastern Europe and other places where restrictions apply [6], [14]. As with COMP128; both of them violated Kerckhoffs' Principle and are both weak. A5/1 was cracked by Biryukov et al [6], [25] and again by [26]. The algorithm can be cracked using a moderate high performance PC only in few seconds. A5/2 also was cracked in less than a day using a method that requires only five clock cycles [22], [24].

SMS Attacks

The SMS's default format is plaintext and this can pose a serious threat in GSM if careful measures are not taken. The SMS can be intercepted during transmission and the information contained in it will then be exposed. The contents can be amended by the attacker who can also execute 'SMS spoofing' by injecting messages into the network with a "spoofed" originator's ID [7]. Also messages can be tampered with at the SMSC where they are stored in plaintext prior to being delivered to their destinations. Encryption in GSM is only applied between the mobile and the base station; but across the rest of the network, no encryption is applied making data vulnerable when intercepted.

Fake Base Station

A fake base station is yet another serious threat posed in GSM communications. The *lack of mutual authentication* and the communication encryption being determined by only the base station are the two major flaws exploited by this attack. Only the subscriber is authenticated to the base station and the latter is not at all authenticated by the former. Also with the base station being in control for encryption; it enables the attacker to choose weaker encryption or none at all [27]. Using a fake base station, one can convince the mobile not to encrypt the communication and without being noticed executes a *man-in-the-middle* attack to make all communications go through him between the caller and the recipient. This way the fake base station can eavesdrop on the entire conversation. Additionally, the fake base station can conduct a chosen plaintext attack on the SIM by sending any RAND of its own choosing and obtain a corresponding SRES. It can thus achieve a SIM attack without even getting physical access to the SIM card.

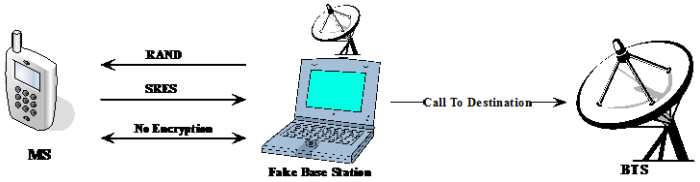

Fig. 5: GSM Fake Base Station

Replay Attacks

By misusing the previously exchanged messages between the subscriber and network, an adversary can arrange for one compromised triple of (RAND, $SRES_{NW}$, *Kc*) to be replayed. This gives the attacker a valid key *Kc* which he can use to impersonate a legitimate user or the network and falsely authenticate a transaction. It can even give a clever fake base station operator the capability to "protect" the communications between the fake BTS and the mobile for no one else to eavesdrop [21].

4 REQUIREMENTS GATHERING AND SPECIFICATIONS

4.1 Current Mobile Banking Systems in Tanzania

A basic study that was conducted in banking and telecommunications industries revealed that most providers use USSD[3] in delivering their services. With USSD; customers can access the financial services regardless of limitations in the mobile phone capabilities. It is a cost-effective technology that is very convenient to the customers. The subscriber initiates the transaction by dialing numbers on a phone screen which constitutes a string that begins with an asterisk (*) sign and terminated by the hash sign (#). Because of its session-based nature; USSD is considered to be relatively more secure as compared to SMS. This is complemented with the fact that no copies of messages are stored either on a customer's mobile or the SMSC. Also the message is encrypted between the USSD gateway and the application server to prevent and misuse. Security concerns with USSD arise from the notion that GSM applies encryption only between the mobile and the base station leaving messages to be carried across the rest of the GSM backbone unencrypted [8], [28]. With cryptographic algorithms used in GSM being broken; data is totally vulnerable when intercepted. Our analysis of the findings from the survey revealed a number of areas most vulnerable for attacks at the *client end*, in the *communication channels*, and at the *server side*. These vulnerabilities can be summarized as the *lack of end-to-end security* for data being transmitted, *vulnerabilities in the authentication mechanisms*, and *vulnerabilities in the application server security management policies*. Authentication issues include the PIN characters not being concealed when typed on the phone making it possible for someone to eavesdrop over one's shoulder. Also the PIN is only 4 digits long constituted with numbers only, thus becoming vulnerable for a *brute force*[4] attack. Confidentiality is also lacking because no encryption is applied to the data between the customer and the server plus no mechanisms for integrity checks are in place. Moreover, the security policies in the application servers leave some vulnerabilities that can be exploited. No periodic changes of PIN are enforced, and some systems allow the use of default PINs.

4.2 Measures to address the identified vulnerabilities

Based on the vulnerabilities identified in our survey; we developed a model with enhanced security controls to address them. These features to be incorporated in our model are summarized in Fig. 6 and discussed below.

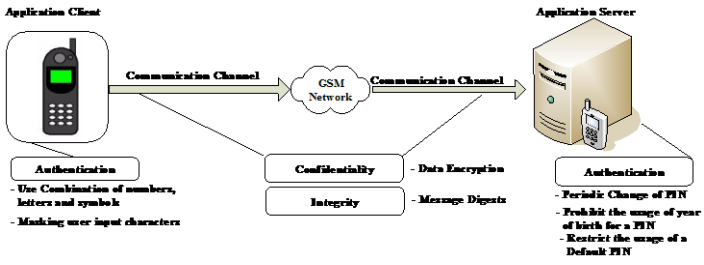

Fig. 6: Security Controls on the client, in the network and the server side

4.2.1 Enhanced Confidentiality of data

Confidentiality of data is enhanced through the use of message encryption to ensure end-to-end security and ensure that data can only be accessed by authorized parties. The encryption processes and the choice for encryption schemes to use are discussed later in this section.

4.2.2 Mechanisms for data integrity checking

Message Integrity is ensured by the use of message digests that are obtained by hashing message contents prior to being transmitted across the network. The message digest is then attached within the sent message. On the receiving end, another digest is generated from the received contents and compared with the attached message digest. If the two digests do not match, the receiver will know that the message integrity has been compromised.

4.2.3 Enhanced Authentication Measures

Authentication is meant to verify users' identities through preventing unauthorized individuals from using system resources by claiming to be someone they are not. The propositions for enhanced authentication include:

a. Having the characters masked as the user types in his/her PIN on a phone in order to conceal it from an eavesdropper who may be watching and later attempt to perform a masquerade attack.

b. We also strengthen PIN strength by allowing the PIN to contain characters other than numbers (a combination of numbers, letters, and symbols is proposed). This will improve the current PIN constituted with 4 digit numbers only. 4 digit numbers produces a maximum of 10^4 (10,000) possible PIN combinations. This is such a small number that can be easily cracked using a brute force attack. A mixture of characters is complemented with imposing a 30 seconds delay between failed attempts and locking the account after 3 consecutive failed attempts.

4.2.4 Improving server security management policies

We also recommend some improvements in the policies that are guiding security management in the application servers. These are mainly associated with the valid character combination that can be used for a PIN and the period of time that a PIN remains valid. The recommended improvements include periodic changing of PIN, restricting usage of default PIN, etc. These are briefly described below:

a. **Enforcing periodic changing of PIN** – Users are likely to disclose their PIN to some other individuals at some point; be it intentional on unintentional (which may involve PIN being intercepted without the user's knowledge). Changing a PIN after a certain period of time is a good security practice to help encounter these cases should the PIN have fall into wrong hands. We enforce in our model the change of PIN after every six (6) months. After this period one must change one's PIN and should not be allowed to perform any transaction on the account other than changing the PIN.

b. **Restricting the usage of default PIN** – The default PIN which is associated with an account when the account is created should be restricted to be used. It

is usually common to many customers and some customers may naively keep using it for their transactions. This makes them vulnerable to attacks should someone with alleged intent get hold of their handsets. Some service providers allow the usage of this system default value. We restrict its usage in our model by forcing the user to change one's PIN before one can do anything on the account.

c. **Prohibiting the use of some character combination as PIN** – Some people tend to use their date of birth (especially the year) as their PIN. This creates a serious vulnerability as related to systems security. Someone's year of birth can be known from different sources; a friend, a family member, student's registration information, employment records information, etc. Using one's year of birth makes it easy for an attacker to perform a fraudulent activity by simply trying the year of birth. The user's year of birth will be recorded during registration and the user should never be allowed to use it as a PIN for the account.

4.3 Specifications and Assumptions

4.3.1 Design Specifications

Secure Message - The message sent needs to be secure in order to preserve confidentiality and integrity of contents between the mobile and the server. Intercepted messages should be unintelligible to unauthorised parties and if the message contents were altered; the recipient should be able to detect the changes.

Low cost and system usability - It is crucial to minimize the number of SMS messages that are sent across the GSM per transaction because each text message sent across the network has a price. The cryptographic processes will add some overhead making the message require more than one SMS. It is therefore necessary to optimize the process in order to minimize the number of SMS messages being exchanged
It is also crucial that the proposed solution be easy-to-use, and user-friendly to its users. The system's user friendliness is a contributing factor for users to abide to the security best practices. Users are likely not to adhere to the security practices if they are too complex. Thus, the authentication and encryption techniques should be carefully selected not to impose too much complication that will affect the system's usability.

Fast Performance - Mechanisms should be implemented to ensure the system's good performance. The transactions should finish being executed in acceptable periods of time. To accomplish this; the server should be relieved of handling any invalid requests that are mistakenly sent to it. Measures should be implemented to ensure that only valid requests are served.

4.3.2 Assumptions

The assumptions that were made in designing the proposed solution are as mentioned below:

a. The model addressed secure communication between the mobile and the application server. It is assumed that the security between the server and the bank databases is handled by the bank itself.
b. It is assumed that customers' handsets are capable of running the client application. Since the application runs on java platform; the mobile phone needs to be java-enabled.
c. It is also assumed that the bank has a secure method to distribute the application to its customers. Be via installation during registration or the application being pushed to a customer over the air; it's up to the bank to decide.

4.4 Choice of Encryption Schemes

In symmetric ciphers; less overhead is involved because only a few SMS messages are exchanged. However, its key distribution is quite a challenge as the two parties must meet in person to exchange a shared key to be used in their communications [29]. With asymmetric key cipher; the overhead is higher because of a number of messages involved. Several messages have to be exchanged between the two parties in order to establish a session key to be used. This renders the asymmetric key crypto to be relatively slower. However, this scheme promises a much easier key distribution since an entity can publish its public key that can be utilized by any other party for secure messaging between them. Choosing one cipher scheme over the other becomes difficult considering their associated strengths and weaknesses. Because of its ease in key distribution; asymmetric cipher is preferable in some instances. But symmetric cipher is also preferred in some cases because of its lower overhead cost and speedy communications. However, in some situations the costs of overhead are not of much importance making asymmetric to remain the favored protocol. In our solution we propose a parallel use of both ciphers to some level. The asymmetric cipher can initially be used for exchanging a shared key between the customer and the bank, and this shared key can then be used in further communications using symmetric cipher. This way both sides will have the same key and can symmetrically encrypt communications between them without ever having to meet. Symmetric encryption is more suitable for use because of the majority of the mobile phones used in developing countries. As opposed to asymmetric ciphers, symmetric cryptosystems require less complex computations which can be accommodated by the low memory capacity handsets used by the majority. The smart phones with highly advanced memory and processing capabilities are not possessed by a large number of users in these communities. Therefore, the message will be symmetrically encrypted prior to being transmitted using a key that is exchanged using asymmetric protocol.

4.5 Passwords and key management assumptions

The following assumptions were made in managing the users' passwords and keys for authentication and encryption/decryption respectively:

a. A registered customer selects a password to be used as a PIN that only him and the bank will know about. This PIN will be required to be changed periodically as specified and one's year of birth cannot be used as a PIN.
b. The encryption key to be used in securing messages between the customer and the bank will be exchanged using a public key mechanism. The application client and the server will both generate a

pair of keys (public key, private key). The public key is publicly known but the private key only the entity knows. Upon request by the customer, the bank will generate a shared key and encrypt it using a customer's public key and send it to the customer. The bank will also associate this shared key with a corresponding customer's account ID. The application client on the customer's phone will use its private key to decrypt the message from the bank and retrieve the shared key. This key will then be used in symmetrically encrypting and decrypting messages that will be exchanged between the customer and the bank. The bank having associated every key with an account ID it will be using the keys to decrypt the messages prior to carrying out a transaction.

5 System Model and Prototype Implementation

5.1 System Modeling

5.1.1 General Structure of the proposed Model

We use an SMS based solution in our model to address the above discussed vulnerabilities. The primary objective is to secure communications between the mobile and the server in order to prevent any tampering on the exchanged data. The reason for adopting an SMS solution is mainly its independency to equipment manufacturers as opposed to the USSD which requires some added features in mobile phones and/or the USSD gateway in order to accommodate the features proposed in the model. This is discussed in details in our paper titled, "*Security perspectives for USSD versus SMS in conducting mobile transactions*" [30]. Our model is structured in three tiers; the *application client* with a java application running on a mobile phone, the *application server*, and a *backend database server* that is integrated in a core *banking system*. These entities are demonstrated in Fig. 7 and discussed hereunder.

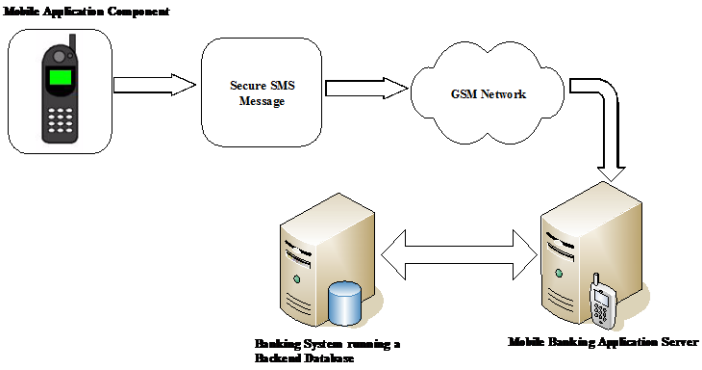

Fig. 7: General architecture of the proposed model

The Mobile Client Application

The client application resides in customer's mobile phone and it is responsible for initiating transaction requests, capture required security details and generate a secure message. The application will then send a message via SMS across the GSM network to a server using a pre-configured SMS short code.

The Mobile Application Server

The server application listens on the network for incoming requests and upon receiving one; it decodes it into some interpretable format. It then uses the designed protocol to perform the security checks on the received request before executing it. This component can be located and administered by a particular application vendor or it can be independently located and administered by the bank itself. It initiates the communication with a banking system that contains a database that has all the banking and security details of the users.

The Backend Database Server

Customer's banking details, security details, and other related user information are kept in the database that is integrated with a banking system. This information is utilized by the server application for customer verification and execution of the various requested transactions.

5.1.2 Structure and general flow of messages

The general structure of the message

The secure message generated by the application client to be sent over the GSM is divided into multiple fields to accommodate the various security details required for our model. The overview of the message structure is as shown in Fig. 8 and explained below. The message is generically formatted with the following fields:

<App Version><Acc ID><Secure Message>

The *App Version* symbolizes version number of the client application that generates the message. It is used by the server application to ensure that only messages generated by valid applications are processed by it. This helps the server to avoid processing unnecessary workloads.

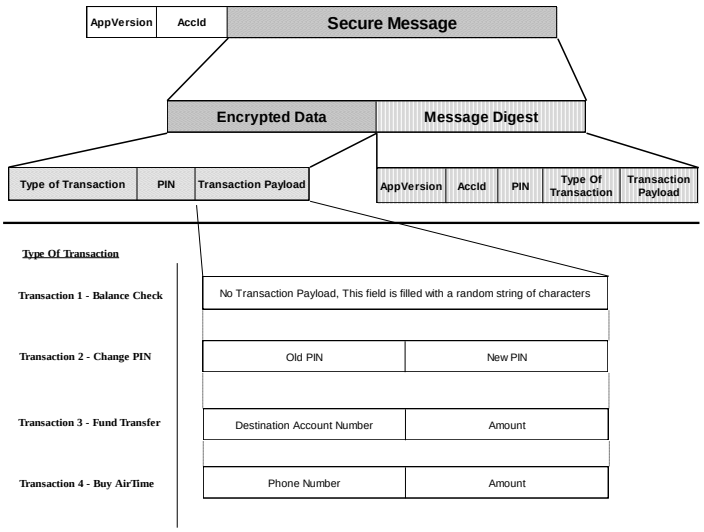

Fig. 8: Message format of the Secure SMS

The *AccID* field contains an account identifier which is sent in cleartext in order to enable checking of customer existence prior to processing of security details. The *Secure Message* field carries the secured contents and is subdivided into two subfields; *encrypted data* and *message digest*. The *Message*

Digest contains the calculated digest of the message that is used by the server for integrity check. In this model; the fields from which a digest value is calculated include *App Version, AccID, PIN, TypeOfTransaction,* and *Transaction Payload*. The subfields that are contained in the *Encrypted Data* field are:

<*TypeOfTransaction*><*PIN*><*TransactionPayload*>

The *TypeOfTransaction* field communicates to the server the choice of the type of transaction requested by the customer. The *PIN* header carries the customer's pre-selected password for authentication on the server. The *TransactionPayload* header contains extra data required for different types of transactions

a. To check an account balance; no extra data is required but the transaction payload field in this type of request is filled with *random characters* for security purposes.
b. A destination *Phone Number* and *Amount* values need to be specified in a transaction payload field of the request to *purchase airtime*.
c. In a request to *change PIN*, the old pin and the new one are required.
d. If the transaction requests for *money transfer*, the *destination account* and the *amount* to be transferred need to be specified.

The overall flow of messages

In a message for transaction request; the *MSISDN*[5] is used as the source address identifying the customer that issues the request. The message will be destined to a server address that will be assigned with a pre-defined *short code*. Going through the SMSC; the message will be forwarded to a *short code* that will be contained in the destination address field. The processes by which a secure message is generated, packaged and sent by client as well as the request receipt and processing by the server are described in the following sections below.

Secure Message Generation and Transmission

The client application on a mobile generates a secure message by capturing the necessary details from the customer and then appends to the message the header fields needed for security checks on the server. Information captured includes *type of transaction* and its associated extra data, the *Account ID*, and the user's *PIN* whose characters are masked to prevent onscreen eavesdropping. After all the details have been captured; the message digest is calculated from the values specified in the above section. Part of these values will be encrypted to ensure that an attacker cannot generate another valid digest if the message is intercepted. The message contents are then encrypted while purposely leaving some fields unencrypted for some preliminary security checks by the server. The *application version* and the *account ID* are left unencrypted in order to allow the server to check if the message comes from a valid application and whether the specified account exists before it decrypts the message. At this stage the message is ready; the preset application version associated with a client application and an account ID are then attached to the secure message and the message is securely transmitted over GSM.

Receiving and Decoding Secure SMS Message

From the cellular network; the message is retrieved by the server and gets processed by first checking if it originates from a valid client using the *application version* header. The message is discarded immediately if the version is invalid and if the message is found to be valid, the account is checked for its existence in the database using the *account ID*. The message is decrypted if the account exists and otherwise; the error message is returned to the customer. The key associated with the account is then retrieved from the database and used to decrypt the message. From the decrypted message; the server uses the same fields used in the client to calculate the message digest by using the same hashing algorithm. The two digests are then compared to check for message integrity. If they match; the message is integral and the PIN is retrieved from the message and compared with the one stored in the database. The PIN is then checked for its validity period and if it is not the system's default value. If the PIN is found to be valid; the requested transaction is carried out and the corresponding response is generated and sent back to the customer. Otherwise, an error message is returned to the customer. The flow chart diagram in Fig. 9 below summarizes the processes involved in server security checking.

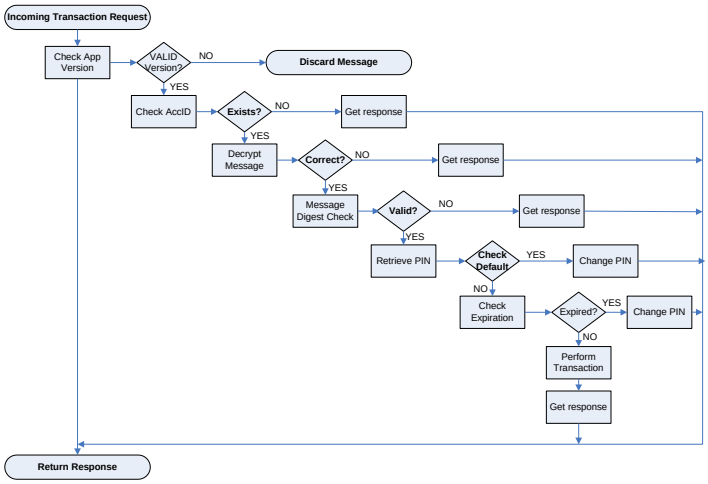

Fig. 9: Flowchart diagram for security checks on the server

5.1.3 Authentication Mechanisms

As it has been mentioned earlier; the current mechanisms for authentication are fairly weak and subjected to guessing and brute force attacks. The four (4) digit numerals used produces 10,000 ($10x10x10x10 = 10^4$) total PIN combinations to be used by hundreds of thousands users of these systems. This implies that several users are certain to share the same passwords and it is more likely for one's password to be guessed which is made even easier by people using their years of birth. Allowing numbers, letters, and symbols adds more possible combinations and makes guessing and brute force become harder to achieve. With this combination (and say using the same 4 character space); we now have 10 numeral digits (0 through 9), 52 alphabetical letters (26 small letters and 26 capitals), and at least 10 symbols (!, @, #, $, %, ^, &, *, (, and) symbols). This makes a total of more than 72 character space domain (10 + 52 + 10 = 72) which when combined in fours produces at least 26 million PINs (72^4 = 26,873,856). Brute force will require a lot of time to be accomplished.

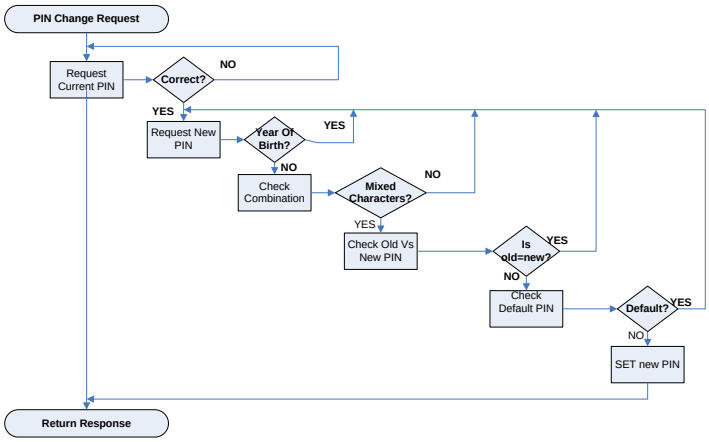

Fig. 10: Security checks in a request to change the PIN

With other restrictions of not using one's year of birth or system's default PIN and periodic changing of PIN; our authentication model is sufficiently secure to resist the attacks aforementioned. Thus, one cannot perform any transaction on the account if one has not changed the default PIN, and if the PIN validity period has expired. These restrictions are realized in the process of changing the PIN whereby in choosing the new PIN; the chosen value should not be a default value, should not be one's year of birth and it must contain a mixture of characters. These restrictions imposed are illustrated in the flowchart diagram in Fig. 10 above.

5.2 Prototype Implementation

5.2.1 General structure of the system prototype

We have implemented a system prototype to demonstrate the various features in our model. Due to time limitations and resources constraints; simulations were used to show how the model would operate in real environment. JAVA Servlets were utilized in the implementation to realize the features incorporated in the model. Servlets are java classes that extend server capabilities in receiving requests and returning appropriate responses to the requesting entity [31]. The general system structure is composed of three separate components; a *mobile banking application client*, the *gateway application* that simulates the SMS gateway (SMSC), and the *application server* that is also running a database server. The client application is implemented by a *Java Command Line Interface* program running on a PC to simulate a mobile phone. The gateway and the server applications are implemented using the servlets and they run on another PC (both can run on the same PC or even two separate PCs). The client cannot directly communicate with the database server but it has to go through the server application. The transaction requests are generated by the user from the client computer and forwarded to the server application through the gateway application. The gateway application servlet receives the requests from the client and relays them to the server and then receives the server responses and relays them back to the client. The server application responds to the client requests by communicating to the bank database server to retrieve and update different stored data. The general setup is as illustrated in Fig. 11 below. Java Enterprise Edition (Java EE) platform was used with *eclipse Juno for web developers* as an Integrated Development Environment (IDE). Eclipse provides the necessary tools for writing java codes for various cross-platform desktop and web applications. The application client was developed as a *Client Application Project* in eclipse and exported as *JAR* files to create executable files for deployment on a target device. The server and gateway applications were both developed as *Dynamic Web Projects* that enable to receive requests via URL and return responses to the requesting entity. For deployment in the target server hardware; these components are exported as *WEB Archive* (WAR) files. *Apache tomcat 7* was used as a *web container* to host and interact with the servlets while the database was implemented using *MySQL* database server. MySQL connection is realized through the *Java Database Connectivity* (JDBC) using MySQL Java connector library. Hypertext Transfer Protocol*, HTTP* is used as a communication protocol between the client and the server. This provides an insecure environment same as GSM in which traffic can be intercepted and viewed by some unauthorized parties. We use *Wireshark* network analyzer to sniff packets of messages exchanged on the network between the client and the server. For data encryption and creation of message digests; the *SECURITY* package included in java was utilized to provide Application Programming Interfaces*, APIs* to be used in cryptography. For data encryption; the Advanced Encryption Standard, *AES* algorithm was used whose simple nature makes it suitable for devices constrained in memory capacities. It is a symmetric block cipher that uses 128 bit blocks for encryption and whose supported key sizes are 128, 192, and 256 bits.

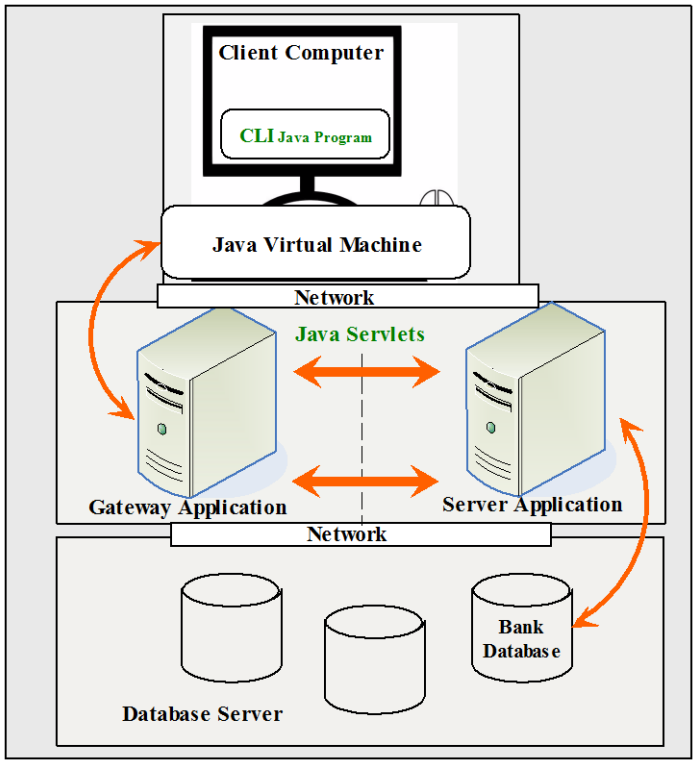

Fig. 11: General setup of the system prototype

Secure Hash Algorithm 1, *SHA-1* was used to create message digests. This hashing algorithm takes a message with a maximum length of 264 bits and produces an output message of 160 bits. The cryptographic operations were implemented in

a class called *SecureMessage* that contains methods for message hashing, data encryption and decryption.

5.2.2 Implementation of features proposed in the model

Message Encryption and Decryption

encrypt() and *decrypt()* methods in the *SecureMessage* class are used for data encryption and decryption respectively. The *encrypt()* method takes in a message string and a key as parameters and returns an encrypted string. It converts the message and the key strings into byte arrays then encrypts the message to produce a ciphered byte array. The result is then encoded into character format using Base64 encoding. The java code below shows the method for data encryption.

```
public static String encrypt(String valueToEnc,        String
encKey) throws Exception {

   keyValue = encKey.getBytes();

   Key key = null;

   try{

     key = new SecretKeySpec(keyValue, ”AES”);

   }catch(Exception e){

          e.printStackTrace();

   }

   Cipher c = Cipher.getInstance(”AES”);

          c.init(Cipher.ENCRYPT_MODE, key);

          byte[] encValue = c.doFinal(valueToEnc.getBytes());

          String encryptedValue = new

          BASE64Encoder().encode(encValue);

          return encryptedValue;
 }
```

The first parameter of the line c.init(Cipher.*ENCRYPT MODE*, key) is set to *ENCRYPT_MODE* indicating that encryption is being performed. The value for the parameter is set to *DECRYPT_MODE* for decryption. AES is used for encryption with 128 bits key which is derived from the encKey parameter in the method's header. This parameter is converted to 128 bits (equivalent to 16 eight bits characters) to match the key length used by the algorithm. The same series of operations are used in a reverse order to decrypt the message. The *decrypt()* method takes in an encrypted string and a key as its parameters, uses Base64 algorithm to decode the message, then uses the key to recover the original message. The java code snippet below illustrates the decryption process:

```
public static String decrypt(String encryptedValue,String
decKey) throws Exception {

  keyValue = decKey.getBytes();

   Key key = null;

   try{

          key = new SecretKeySpec(keyValue, ”AES”);

          }catch(Exception e){

                 e.printStackTrace();

          }

   Cipher c = Cipher.getInstance(”AES”);

   c.init(Cipher.DECRYPT_MODE, key);

   byte[] decodedValue = new

BASE64Decoder().decodeBuffer(encryptedValue);

   byte[] decValue = c.doFinal(decodedValue);

   String decryptedValue = new String(decValue);

   return decryptedValue;

}
```

Message digests are calculated using the *createDigest()* method in the *SecureMessage* class. The method takes in the message as a parameter and then obtains an instance of a hashing algorithm; we have used SHA-1 in our demonstration. The message is first converted into bytes from which a digest value is calculated. The resulting byte stream is converted into hexadecimal format then a string of the hashed value is returned using java string conversion methods. Below is the code used to calculate message digests.

```
public static String createDigest(String valueToHash){

   String digestValue = null;

   MessageDigest md = null;

   try {

          md = MessageDigest.getInstance("SHA-1");

          } catch (NoSuchAlgorithmException e) {

                 e.printStackTrace();

          }

   byte [] preparedMsg = null;

   byte [] hashedValue = null;

   preparedMsg = valueToHash.getBytes();

   md.update(preparedMsg);

   hashedValue = md.digest();
```

```
  //convert the byte into hexadecimal format

  StringBuffer hexString = new StringBuffer();

    for (int i=0;i<hashedValue.length;i++) {

            String hex=Integer.toHexString(0xff &
hashedValue[i]);

            if(hex.length()==1){

            hexString.append('0');

            }else{

                hexString.append(hex);

            }

    }

    digestValue = hexString.toString();

    return digestValue;

}
```

Application Connection to the Database
MySQL Java library is utilized to provide the JDBC needed for communication of the application with the database. In setting up a connection; a URL containing the location of the MySQL to Java connector driver need to be specified. Its general format is "jdbc:mysql:///(Path for database location on disk)" as shown in the code below; a *username* and *password* is used to control access to the database server.

```
//Accessing a driver from the JAR file

Class.forName("com.mysql.jdbc.Driver");

//Connecting to a specified database URL

dbcon=DriverManager.getConnection("jdbc:mysql://localhost:3
306/banking", "username","password");
```

Database Schema
A MySQL database to handle the application data in our demonstration consists of six tables whose schema is presented here:

Account(AccId, AccBal, CustomerId)

Customer(CustomerId, FName, LName, YearOfBirth, CipherKey)

Phone(PhoneNum, PhoneBal, CustomerId)

Pin(Pid, PinNum, InitialDate, CustomerId)

Staff(StaffId, FName, LName, UserName, Password)

Transaction(TransacID, CustomerId, AccId, amount, TypeOfTransaction)

The a*ccount* table is used to hold account information for registered customers. The table *customer* stores customer information and contains a key for ciphering mechanisms. The *phone* table stores mobile phone numbers and it is related to the customer table with a *CustomerId* field. *Pin* table contains the PIN details for the customer with their respective initial dates in order to manage their valid duration of use. The system administrator's details for controlled access to server application are kept in a *staff* table. And for the purpose of keeping records of the transactions that have been carried out on the system; the *transaction* table is used.

5.2.3 System Output Results

For demonstration; two versions of the application were developed to show the difference; one with no security features incorporated and another with the features incorporated. The messages sent across a network by the two versions were analyzed by a network analyzing tool called *Wireshark* (http://www.wireshark.org/). This tool was used to capture the packets that were flowing on the network from a client computer to the server. In the first version of the application where no securities were used; the message can be intercepted and its contents viewed by someone sniffing on the network. Wireshark can be used to sniff packets from any source to any destination using a device connected to that same network. Fig. 12 below shows a captured message that was sent from *172.16.5.48* source address to a destination server with *172.16.3.89* IP address. Since the message is sent in a clear; its contents can be seen by observing the URL string *http://172.16.3.89:8080/BankingServerApp/AppServer?appVersion=BankApp-Vrsn1.0&accId=18201062334&message=TRN,dudu8791,101200021300:45000.0***.** *8080* is a port number identifying *Apache tomcat web container, BankingServerApp* is the name of the server application, *AppServer* is the name of the file on the server that processes the received request and *BankApp-Vrsn1.0* is an application version number. *accId=18201062334* shows the account number on which the transaction is being performed. The remaining parts show the specific details of the transaction. In *message=TRN,dudu8791,101200021300:45000*; *TRN* means that the type of transaction being requested in *TRANSFER*, *dudu8791* is the customer's PIN, *101200021300* is the destination account number and *45000.0* is the amount being transferred. A lot of damage can be done if this information falls into wrong hands. The intercepted message can be modified for various malicious intents; one may change the destination account ID, or the amount being transferred or even both. Since the PIN is also seen, it can later be used to execute masquerading attacks on the account.

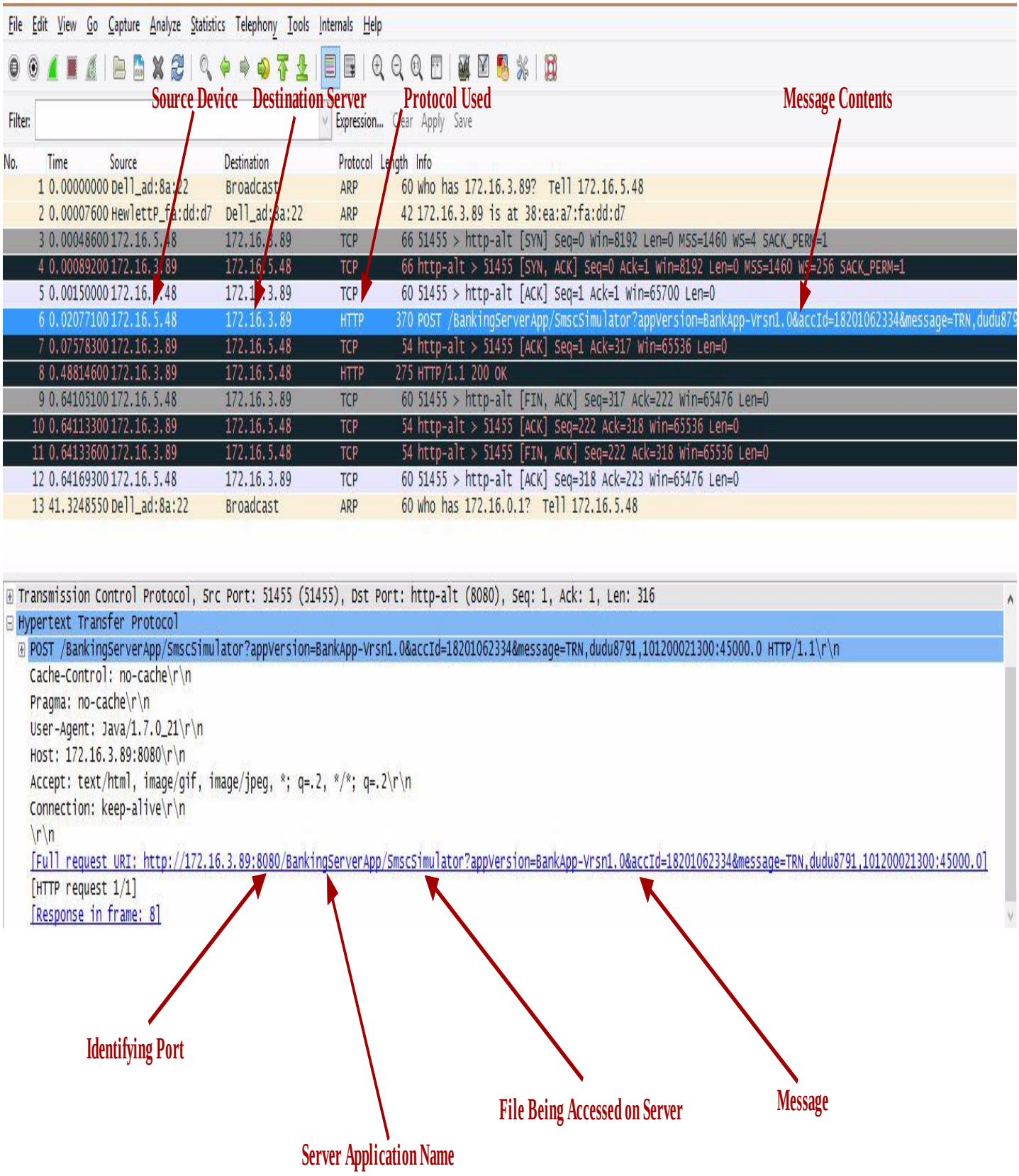

Fig. 12: Wireshark screenshot for a message captured from an insecure application

In the second version with proposed security features implemented, the message is secured before being sent on the network. Fig. 13 below shows a snapshot of the captured message that was sent from the same source address to the same destination as before. This content is seen on the snapshot as *http://172.16.3.89:8080/BankingServerApp/AppServer?appVersion=BankApp-Vrsn1.0&accId=888&encMsg=7Kd%2FgMwi%2FTayYIHMRF%2FmMVQm%2BI%2Fysd%2FXYMbvwQKfuRA%3D&digestValue=0b5a5fa8eae4e3bfb9ecd456a0499b485cca20*. *8080, BankigServerApp, AppServer, appVersion and accId* elements have the same meanings as before but in this case the transaction details have been packed in an encrypted string, *encMsg*. This element contains only a string of random characters *7Kd%2FgMwi%2FTayYIHMRF%2FmMVQm%2BI%2Fysd%2FXYMbvwQKfuRA%3D* . The hashed value of the message is also attached here for integrity validation. This is contained in element *digestValue* whose value appears as *0b5a5fa8eae4e3bfb9ecd456a0499b485cca20*. These two elements; *encMsg* and *digestValue* bear no useful meanings to an attacker upon interception. This ensures information security even in the incidence of falling into wrong hands. Application version and the account id are purposely left unencrypted for reasons explained in *section 5.1.2* pertaining the, *"general structure of the message".*

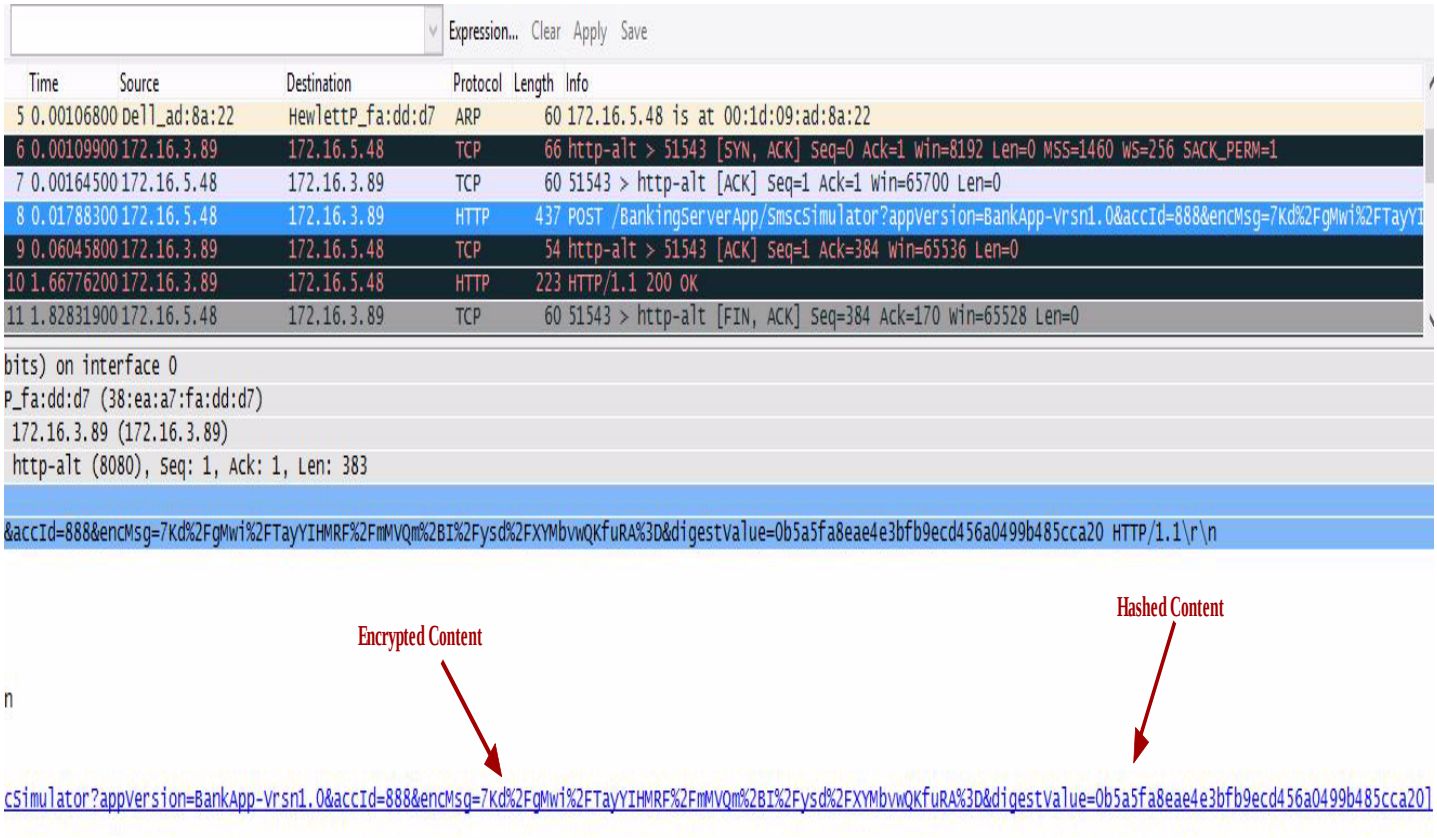

Fig. 13: Wireshark Interface capture for a message from a secure application

6 SUMMARY AND DISCUSSIONS

In this study; an SMS based solution has been designed and its prototype has been implemented to address the security issues that are associated with conducting mobile transactions over GSM. For enhanced security measures in mobile transactions; data need to be encrypted and provisions for ensuring message integrity have to be enforced prior to sending the message over the network. This is for the purpose of ensuring end-to-end security of data between the customer and the financial institution. The SMS solution requires a java application be installed on a customer's phone to capture the required security details and generate the secure message. When the server receives the request it performs all the security checks and executes the requested transaction when all the checks are valid.

6.1 Security analysis of the proposed model

The main focus of our design goals for the model was to conform to the *principles of secure service*. These are user *authenticity*, *confidentiality* of data, *non-repudiation*, data *integrity*, and *availability* of service [32]. We give a brief description below of how these services are addressed in our model.

User Authentication

The user is authenticated to the system using a preselected PIN which a user must give while performing any transaction on the system. Some improvements have been added by using a combination of numbers, letters and symbols to increase resistance against brute force attacks. The enforcing of periodic PIN change and the use of encryption keys also ensure that only the authorized customer can perform a transaction.

Data Confidentiality

Symmetric key ciphers are used to encrypt message contents in order to ensure data confidentiality. Since the keys used are known by the customer and the bank only, the communications between them will remain confidential as long as the keys remain a secret.

Non-repudiation

The key used for encryption/decryption is uniquely associated with only one subscriber. Since only this key can encrypt messages that will be successfully decrypted by the server, neither of the parties can deny its involvement in any transaction. Only the customer and the bank should have knowledge of the key; all successful transactions, therefore, must have originated from a customer with a correct key. Thus, the encryption key in addition to the PIN can be used to hold a customer responsible for transactions performed on one's account.

Integrity

Message digests are used to ensure message integrity where hashes of message contents are calculated at both ends and then compared. If the digest calculated by the sender differs from that generated by the receiver; the recipient will detect a compromise in the message integrity.

Service Availability

The system's availability will largely be influenced by the network operator's availability. It will eventually be down if the operator's network is down. However, mechanisms are enforced to ensure optimal server performance. The application is capable of multi-processing as much transactions as the server hardware can handle. Developers' choices of server hardware will determine the number of transactions to be simultaneously processed by the server. Also as described in previous sections; the *application version* attached to the transaction request message helps the server discard any messages that have been misaddressed to the server. In the long run, this helps to prevent *denial of service* attacks on the system. Table 2 below summarizes the comparisons between the existing systems and our model in security perspectives.

Table. 1: Security comparison between the existing approaches and our proposed model

Existing Approaches	***Our Proposed Model***
The current USSD technology sends its data in clear text across GSM	Our model utilizes SMS based solution that encrypts data between the client and the server
No mechanisms for message integrity checking in existing implementations	Messages are checked for integrity at the receiving end using message digests.
The current PIN is only four digit numbers	We use a combination of numbers, letters, and symbols to construct a PIN.
The PIN in current implementations can be used indefinitely without expiring.	We implement mechanisms for users to change their PINs periodically.
The PIN characters are visible on screen as the user types in the password.	The java application installed on the phone conceals the character inputs for a PIN.
Some of the current implementations allow the usage of a system's default PIN.	We restrict users to use the default PIN and they must change it before they can access any service.

7 Conclusion and Future Works

An enhanced security model for mobile banking systems has been presented in this paper. The model incorporates enhanced security controls to be used in securing transactions; the study area being Tanzania to represent many of the developing countries. From the study; the requirements for the proposed model were identified, the model was then designed and its prototype has been implemented. Mechanisms have been incorporated into the system to protect data as it is transmitted across 2G GSM networks and ensure secured mobile transactions. Despite the release of more secure 3G systems which address most of the 2G security weaknesses, the backward compatibility with 2G systems poses a great security risk. The 3G connectivity is only accessible in areas around big cities, townships and some strategic remote areas. However, most of the rural places still use GSM for connectivity. Thus, it is important for financial service providers to protect their communications to and from customers in order to ensure secure conductance of mobile transactions. Our model serves this purpose by enforcing a number of security controls to the existing systems; the messages are encrypted, and hash codes are used to ensure message integrity. The policies in the server security management have been improved as related to the usage of default PIN which has been restricted, and enforcement of periodic changing of PIN. The model can thus be used to securely conduct mobile transactions over unsecured channels. The focus for our future works is to implement and test our model in real environment, and to explore on feasible ways to improve security of transactions carried by USSD technologies. Also a feasibility analysis needs to be conducted on the use of future mobile communication systems for conducting mobile transactions along with its security implications.

References

[1]. mobiThinking. Global Mobile Statistics 2013. 2013 [cited 2013 20 August]; Available from: http://mobithinking.com/mobile-marketing-tools/latest-mobile-stats/a#subscribers.

[2]. TCRA. Telecommunications Statistics March 2013. 2013 [cited 2013 20 August]; Available from: http://www.tcra.go.tz/images/documents/telecommunication/telecomStatsMarch13.pdf.

[3]. IST-Africa. Introduction - Republic of Tanzania. 2010 [cited 2013 20 August]; Available from: http://www.ist-africa.org/home/default.asp?page=doc-by-id&docid=4324.

[4]. Wikipedia. Mobile Banking. 2013 [cited 2013 10 September]; Available from: http://en.wikipedia.org/wiki/Mobile_banking.

[5]. Emmanuel, A. and B. Jacobs, Mobile Banking in Developing Countries: Secure Framework for Delivery of SMS-banking Services. 2007, Radboud University Nijmegen.

[6]. Biryukov, A., A. Shamir, and D. Wagner. Real time cryptanalysis of A5/1 on a PC. in Fast Software Encryption. 2001: Springer.

[7]. Kaur, G., P. Kaur, and K.K. Saluja, A Review of Security issues and mitigation Measures in GSM. International Journal of Research in Engineering & Applied Sciences, 2012. **Volume 2**(Issue 2 (February 2012)): p. 16.

[8]. Toorani, M. and A. Beheshti. Solutions to the GSM security weaknesses. in The Second International Conference on Next Generation Mobile Applications, Services and Technologies, 2008. NGMAST'08. 2008: IEEE.

[9]. Chikomo, K., et al., Security of mobile banking. University of Cape Town, South Africa, Tech. Rep., Nov, 2006. **1**.

[10]. Wamyil, M.T. and M.B. Mu'azu, Gsm Networks: A Review Of Security Threats And Mitigation Measures. Information Manager (The), 2006. **6**(1): p. 16-24.

[11]. UNCTAD. Mobile Money For Business Development in the EAC, A Comparative Study of Existing Platforms and Regulations. in UNCTAD/DTL/STICT/2012/2. 2012: United Nations.

[12]. O'Brien, K.J. Cellphone Encryption Code Is Divulged. 2009 [cited 2013 15 August]; Available from: http://www.nytimes.com/2009/12/29/technology/29hack.html?pagewanted=all&_r=1&.

[13]. Van der Merwe, P.B., Mobile Commerce Over GSM: A Banking Perspective on Security. 2003, University of Pretoria.

[14]. Chong, M.K., Security of mobile banking: Secure SMS banking. Data Network Architectures Group. University of Cape Town, South Africa, 2006.

[15]. Chikomo, K., Mobile Banking Security using GPRS, in Computer Science: Data Networks Architecture Group. 2006, University of Capetown: Cape Town.

[16]. Chong, M.K., Usable authentication for mobile banking, in Computer Science. 2009, University of Cape Town.

[17]. Bilal, M. and G. Sankar, Trust & Security Issues in mobile banking and its effect on customer, in School of Computing. 2011, Blekinge Institute of Technology: Karlskrona. p. 63.

[18]. Panjwani, S. and E. Cutrell. Usably secure, low-cost authentication for mobile banking. in Proceedings of the Sixth Symposium on Usable Privacy and Security. 2010: ACM.

[19]. Narendiran, C., A new approach on secure mobile banking using public key infrastructure. International Journal of Computing Technology and Information Security, 2011. **Vol.1**(No.1): p. pp.40-46.

[20]. Zhang, F., Secure Mobile Service-Oriented Architecture, in Information and Communication Technology. 2012, Kungliga Tekniska Hogskolan Royal Institute of Technology.

[21]. Stamp, M., Information security: principles and practice. 2nd Ed. ed. 2011, New Jersey: John Wiley & Sons Inc.

[22]. Paik, M. Stragglers of the herd get eaten: Security concerns for GSM mobile banking applications. in Proceedings of the Eleventh Workshop on Mobile Computing Systems & Applications. 2010: ACM.

[23]. Briceno, M., I. Goldberg, and D. Wagner, A pedagogical implementation of the GSM A5/1 and A5/2 "voice privacy" encryption algorithms. 1999.

[24]. Barkan, E., E. Biham, and N. Keller, Instant ciphertext-only cryptanalysis of GSM encrypted communication, in Advances in Cryptology-CRYPTO 2003. 2003, Springer. p. 600-616.

[25]. Biryukov, A. and A. Shamir, Real time cryptanalysis of the alleged A5/1 on a PC. 1999.

[26]. Ekdahl, P. and T. Johansson, Another attack on A5/1. Information Theory, IEEE Transactions on, 2003. **49**(1): p. 284-289.

[27]. Kröger, C., GSM security, in 14th Twente Student Conference on IT. 2011: Enschede, Netherlands.

[28]. Krugel, G.T., Mobile Banking Technology Options. FinMark Trust, 2007.

[29]. Lockefeer, L., E. Hubbers, and R. Verdult, Encrypted SMS, an analysis of the theoretical necessities and implementation possibilities. 2010.

[30]. Nyamtiga, B.W., Security Perspectives for USSD versus SMS in conducting mobile transactions: A case study of Tanzania. International Journal of Technology Enhancements and Emerging Engineering Research, 2013. **submitted for publication.** (Pending publication).

[31]. Wikipedia. Java Servlet. 2013 [cited 2013 29 August]; Available from: http://en.wikipedia.org/wiki/Java_Servlet.

[32]. Stallings, W., Network security essentials: Applications and Standards. Third Edition ed. 2007: Pearson Education India.