International Journal of Security and Its Applications
Vol.8, No.5 (2014), pp.161-184
http://dx.doi.org/10.14257/ijsia.2014.8.5.16

A Novel Approach to Identify a Fraud Website Using Android Smartphone under the Collaborative Frameworks of QR Codes and GPS and Motion Parameters of the User

Soham Sengupta[1], Dr. Deabsree Chanda Sarkar[2], Dr. S. Biswas[3] and Prof. ParthaPratim Sarkar[4]

Department of Information Technology, JIS College of Engineering
& Research Fellow, DETS, University of Kalyani
soham.sengupta.java@gmail.com
[2,3,4]DETS, University of Kalyani

Abstract

Use of personalized security mechanisms among financial sectors is gaining rapid momentum day-by-day. Banking and e-shopping portals, which are paramount of cyber-attacks, strongly recommend that both the merchant (i.e., a merchant web portal) and its customers (customers using these portals) be certain about each other's identity. This emphasizes not only that the merchant portal must be able to detect an attacker spoofing the identity of one of its customers, but also that a customer must not leave her secrets with a fraud-cum-look-alike website spoofing address of the merchant's portal.This thesis envisages a novel, scalable approach to detect a fraud, look-alike web page to help a customer unaware of digital certificates, Internet security policies and their glitches, truly recognize her merchant's web-site using her smartphone. The approach uses a reverse challenge-response framework; and uses QR codes which are generated dynamically and depend on the GPS parameters of the customer. The customer uses her smartphone to scan the QR codes with an application provided by the merchant; which detects correctly whether she logged on to the genuine website. The additional benefit of this approach is that it can be modelled to offer a novel, non-telephonic two-step authentication system with minor modifications.

Keywords: *Personalized security, GPS Encryption, QR codes, Android, Accelerometer & gyroscope, Two-step authentication*

1. Introduction

With a concerning rise in the number of cybercrimes, financial sectors have been adopting personalized security measures to protect themselves as well as their customers. Though cyber-attacks on a banking or financial web site commonly refer to a hacker (*e.g.,* a software or a person) which spoofs the identity of a valid customer, sometimes valid customers are fooled by and leave their secrets with a fraud portal that spoofs the identity or is a look-alike of the merchant's or bank's web site.The most common ways that these fraud web sites follow to fool a merchant's customers, involve spoofing the identity of merchant's web site. This can be achieved by deploying a DNS attack. A compromised DNS can cause a customer to get a fraud web site with her browser request. [Figure-1] illustrates how a wrong DNS entry might fool a customer. Sometimes, a misspelled host name, or misspelled web site search-results may forward the user to an unintended site with a similar domain name. An unethically developed web browser, or sometimes a plug-in extension to a browser may redirect a user to a malicious

ISSN: 1738-9976 IJSIA

web site instead of the intended portal. An unofficial release of an open source browser that might have been developed to display an untrusted web site as trusted, and for phishing sensitive user information, may sometimes cause a damage to the user's banking system. [Table-1] lists some common approaches that attackers frequently adopt to steal user information from an e-commerce or banking portal.

This thesis, however, aims at detecting a fraud web page, which a user might have mistaken for the genuine web page by some means or other. The solution envisaged by the authors suggests a mobile application that scans a QR code on the web page(s) to detect if it is a genuine one. The QR codes generated by the server of the organization dynamically varies from one user to another, the surfing timestamp and the GPS location of a user. This 3-way security architecture ensures that the QR codes cannot be replicated by an attacker web server and hence never gets through the verification process by the mobile application. A thorough case study of the fraud web page detection architecture, its mobile-and-web integration techniques, accompanied by a performance analysis with an android application deployed on a variety of smart phones have been presented in this article. Also, the thesis shows how this architecture is equally good to be an excellent substitute for the existing 2-step authentication system, without requiring any telephony (*e.g.,* a One-time password sent as an SMS or over an automated voice call to the registered mobile number of the customer).

2. Proposed Model

Before we discuss the architecture of the proposed model, the thesis prescribes to have a look at the notations and mathematical operators that have been used throughout and enlisted in [Table-2]. The architecture assumes that an Organization, a bank for example, provides its customers with a dedicated mobile application for their smartphones. The application and a web service on the bank's portal share some common, secret knowledgebase as discussed in [Table-3]. After a customer installs the application on her smartphone, she must register the application with the bank's web service so that it authenticates the user and download some additional knowledgebase from the web service, which can detect a fraud web page apparently belonging to the bank's portal.

2.1. Registering the Mobile Application

The mobile application must be registered with the bank's web service with the customer's Internet banking credentials before it can be used to detect a fraud web page. The process of registering the mobile application is illustrated in [Figure-2]. The steps involved in the registration process are discussed below:

a) The customer logs in to her Internet banking service through the mobile application

b) The server (web service) authenticates the customer by her Internet banking credentials

c) The server (web service) maintains a pair of asymmetric keys dedicated solely to the proposed architecture.

d) The server (web service) carries out the following computations:

i. Fetch the current timestamp from the system (T_1:= current Timestamp)

ii. Calculates a value $q=f(T_1)$

iii. A one-time key is created by encrypting the public key of the server with the value calculated in the previous step. $K_S := E(K_{+Server}, q)$

iv. Calculate a digest number U_1 from the various account information of the customer

v. U1 is encrypted with the private key of the Server. $U_1' := E_{-Server}(U_1)$

vi. An application data packet containing the timestamp, the one-time key and the encrypted account digest is sent to the mobile application.

e) The application data packet received by the mobile application carries out the following computations:

i. The public key of the server is found out as

- $q: = f(T_1)$
- $K_{+Server} := E(K_S, q)$

ii. The account digest number of the customer is decrypted using the public key calculated above:

$U_1 := E_{+Server}(U_1')$

iii. The mobile application now populates its local application database with the public key of the server and the account digest number of the customer.

The registration process of the mobile application is tabularized as Algorithm 1 in [Table-4]. The registration process is one-time. Once the application has been registered, it can be used to detect a fraud web page by following the mechanisms discussed in the section [2.2]

2.2. Verification Process to Detect a Fraud or Genuine Web Page

The portal of the bank is so developed that all its web pages, where the customer is required to provide her information (*e.g.,* bank account number, a transaction number, debit card number, PIN *etc.,*) must show up a QR code which the server generates dynamically based on the customer's identity. The customer uses her mobile application, which is already registered with the web service as discussed in [Sec. 2.1], to scan this QR code; and the application verifies and tells the customer whether the webpage she is logged on to, is a genuine or fraud one.

The process of generating the QR code that is shown on the bank's web page is depicted in Algorithm 2 [Table-5] and Algorithm 3 in [Table-6] illustrates how the mobile application authenticates a web page. We have ensured that a fraud web service cannot generate a QR code which gets through the verification steps by the mobile application unless all of the constraints mentioned below are satisfied:

a) Definition of the timestamp function, *i.e.,* f(t) is exposed to the attacker server.

b) The account information digest (U_1) is known to the attacker web service. Since the generation of the QR codes depends on a customer's account digest number (U_1), it is infeasible for the attacker to get hold of an arbitrary account's digest number, because the server never knows

i. The full account information of an arbitrary customer.

ii. Which of the columns of the customer account table are used to calculate the digest number

iii. The digest algorithm itself

c) Finally, the private key of the genuine server ($K_{-Server}$) is known to the fraud web service.

2.3 Analysis of Vulnerability

This section presents a comprehensive case study analyzing the vulnerability of the proposed architecture. We guessed of possible adverse circumstances where pieces of security information (*e.g.,* the timestamp function, a customer's account information, the Digest Algorithm *etc.,*) were assumed to have leaked out to the fraud web service; and

show how the architecture survives the adversity and still holds foolproof and the mobile application still detects a fraud web page.

2.3.1 The Definition of the Timestamp Function has been known to the Attacker:

Under this situation, let us assume that the attacker web site produces a QR code at timestamp $t=t_2$

a) It generates a pseudo-random number (R)

b) Calculates the timestamp value i.e. $q=f(t_2)$

c) Makes a brute force attempt to guess the account digest number of the customer. If the original digest number (U1) of the customer is taken as a 32-bit integer, the probability that the brute force attack to guess the digest number succeeds is = 2^{-32}

d) The attacker web service must also make a brute force attack to guess the private key of the bank's web service, which is , let us say, a 512-bit integer. So, the probability of making a correct guess is = 2^{-512}.

Finally, the probability of fulfilling the above conditions is = 2^{-512} x 2^{-32} = 2^{-544}. With a dedicated attacking hardware, it will require around four weeks to succeed with brute force attack. But by then, the timestamp would have become changed (too old to be accepted by the mobile application).

2.3.2Attacker knows Customer's Account Digest as Well as the Timestamp Function:

Knowing the customer's account digest number in addition to knowing the timestamp function will reduce the attacker web server's efforts by the order of 2^{32}. And with an attacker dedicated for one specific customer will have probability of 2^{-512} to successfully represent itself as the bank's portal. This probability is too little and we conclude that the architecture proposed in this thesis is almost foolproof.

2.3.3 Attacker has User-account Digest, Timestamp-function and Bank's Private Key:

This is the extreme of a scenario and should be realized by ideally a super attacker only. Though it never happens, an attacker web service, in such a situation, can successfully make its web pages pass the authentication by the customer's mobile application by generating a correct QR code. Hence, we incorporate some additional mechanism in the existing architecture to make it more dynamic and less penetrable. The customer's location parameters (GPS) obtained with her smartphone together with her movement parameters obtained with Accelerometer and Gyroscope are introduced to make the detection system more robust.

2.4 Introducing GPS and Customer's Movement Parameter

The mobile application periodically collects the phone's *GPS* location parameters (*e.g., latitude, longitude etc.,*), along with the parameters representing its linear movement in the three-dimensional space; stores these parameters as local application data and sends them in any predefined permutation to the bank's web server. The bank's web service uses these additional parameters to generate the QR codes and the mobile application uses them in the *reverse-computational* techniques to detect a fraud web page. [Table-7] presents *Algorithm 4* which discusses formation of a QR code by the bank's web service

with location and movement of the customer introduced in this architecture. This *algorithm* applies equally to smartphones both with and without support for accelerometer and gyroscope, because all the smartphones might not offer these functionalities. Incorporation of the customer's location and movement parameters in the technique of *QR* code generation requires little bit of amendment.of the functionality described in *Algorithm 3* [Table-6]. The modified functionality of how the mobile application detects a fraud web page is illustrated in *Algorithm 5* presented in [Table-8]. Amalgamation of these additional parameters (*i.e.,* location parameters and movement parameters) makes the architecture more dynamic and hence, less vulnerable to brute force attack.

2.5 Performance Analysis of the Implementation of the Architecture

The solution envisioned in this thesis to detect a fraud web page has been implemented for Android platform by the author and this section presents a detailed performance analysis of the web-to-mobile integration process with genuine test results.

2.5.1 The Android Application: The mobile application implemented in compliance to the proposed architecture, has a minimum platform requirement of *Android* version *2.3.3*. The device must have a Camera and support of *GPS*/*A-GPS*. Though the envisaged model offers best security on devices with Accelerometer Hardware support, it is compatible with low-end devices which have no support for motion sensing offered by *accelerometer* and *gyroscope*. [Figure-3] illustrates the application screenshots of the one-time registration process. [Figure-4] presents a screenshot showing a fraud web page detected by the application. The web integration of the proposed architecture is presented in [Figure-5]. The next section analyses the algorithms, their performance on the android device with relevant data.

2.5.2 Analysis of Algorithms: So far, we have not discussed anything about the common knowledgebase that the banks' web portal and the respective mobile application share. As presented in [Table-3], the shared knowledge contains different functions which we are yet to focus on. These hitherto unfocused functions and their respective parameters are:

- The ***Time Stamp Function*** , ***f*(ts)**
- The common symmetric encryption technique ***E(****data****,*** *key****)***
- The asymmetric cryptographic algorithm ***E +Server (data)*** or ***E –Server (data)***
- The location function, ***G***(*latitude, longitude, altitude*)
- The motion function, ***Ψ***(*Fx, Fy, Fz*)

We do not define these functions concretely, because these are supposed to be specific to an implementation and the thesis aims at providing a *generalized* architecture to detect a fraud web page by dint of a smartphone application. However, the implementation of the architecture by the authors used ***DES*** as a symmetric encryption technique and ***RSA*** as the asymmetric cryptographic approach. A sample definition for the time function is shown in [Table-9]. The location function and the motion function must produce a unique value for a given set of parameters; and a very small variation in at least one of the parameters will yield a considerably different and unpredictable value. This means that two places geographically located very close to each other will have considerably unpredictable and different values produced by the location function. Also, the motion function will ensure that very similar motions must not produce the same or a close value. [Table-10] presents a sample location function, while [Table-11] suggests a simple motion function. The combination of all these ensures that the attacker cannot generate a

successful QR code to get through the verification mechanism by guessing and forging the location and motion of the customer.

2.6 Scope of the Proposed Architecture to Future Research

The thesis, apart from offering a novel solution to detect a *fraud* or *forged* web site, cues a novel approach to substitute the traditional, telephonic 2-step authentication systems used by banking portals and many non-financial web services (*e.g.*, Gmail *etc.*,). The idea behind a two-step authentication systems is that:

Bob might have stolen e-mail id and password of *Alice*, who does not know that her login credentials are now with another person. Using the traditional password authentication mechanism, both Alice and Bob will be able to use Alice's e-mail account. In this, not only the e-mail privacy of *Alice* is compromised to *Bob*, but she can be denied to login to her own e-mail account if Bob has already changed the password.

To address this glitch of password authentication system, the 2-step verification mechanism has been introduced, requiring *Alice* to link her mobile phone number with her account. When the user, with the 2-step verification system enabled on her account, attempts a log-in with her login credentials, the server will send a random challenge, often a random number commonly known as an *OTP* (One-time password), to her registered mobile number as a text *SMS*, and the challenge remains valid for a predefined interval. If the user who had attempted the login were indeed the genuine user, she must have her mobile phone with her and would have correctly responded to the challenge arriving on her mobile phone as a text *SMS*. Generally three successive incorrect attempts lock the account temporally to prevent further spells of an apparent brute-force attack by someone (a person or a Software) who has come to know the user's password but is yet to steal her mobile phone.

A severe drawback to this system is often felt when the respective mobile operator experiences excessive network load, causing delayed arrival of the messages. If the SMS arrives after the validity of the one-time challenge has expired, a genuine user is denied to log-in and successive correct responses to expired challenges may block the account.

To cope up with this issue, automated Voice Calls have been substitute to SMS. Still, this has an added disadvantage which was not with SMS. When the user attempts a login and server is about to generate an automated voice call to her mobile number, a friend or relative of hers might already have initiated a call to her mobile. The server, in this case, gets the user's phone number busy and aborts the calls. While an SMS and a voice call can be received simultaneously by a mobile unit, because a voice call and a text message use different channels; two different calls cannot be received simultaneously. One of the calls must be on waiting while another is in progress. But a call cannot be initiated when another has already been connected but not received. The call-waiting facility, hence, does not help here.

The authors propose a far better solution to substitute of this two-step telephonic authentication, using the same architecture described in this thesis. The solution requires *neither* an automated voice call *nor* an *OTP* sent as SMS. This simply works without any telephony as described below in the next section.

2.6.1 A Novel Two-step Authentication Mechanism: Functionality of our new 2-step authentication is as follows:

a) The user will download the 2-step authentication application on her smart phone from a trusted application store or the bank's (or the respective organization's) web site.

b) The user must log-in to the web service (*e.g.*, bank account *or* E-mail account *etc.*,) using the application and get her smart phone app registered and linked to her account.

This step is very much similar to *Algorithm 1* presented in [Table-4]. This is a one-time setup only.

c) Once the app is registered, the user can attempt a log-in to her web account using a browser.

d) Instead of using an SMS or an automated voice call, the server presents a QR code on the next web page. The QR code will contain an encrypted *One-time challenge* (*OTC*), which can be decrypted by only the mobile application registered and linked to the user's account.

e) The process of generating the challenge by the web service is similar to *Algorithm 2* presented in [Table-5].

f) The process of generating the response to the *One-time challenge* is similar to *Algorithm 3* presented in [Table-6].

Algorithms 6, 7 and 8 presented in [Table-12] describe the entire process of the proposed architecture of a two-step authentication system.

2.6.2 Android Implementation of Proposed 2-step Authentication Model: The *2-step* authentication mechanism, coming up as a side work of the thesis, has been implemented and tested across various smartphones running on Android platform. [Figure-6] shows the screen shots relevant in this context. This process of two-step verification *or* authentication is not only much faster but also very effective in the following aspects:

- It does not require the server to use *Short Messaging Service* (SMS) and the user do not need to wait till arrival of the SMS containing the *OTP*. The load on the mobile network, as well as the *server messaging queue* on the SMS gateway, is reduced; and the cost of the system becomes absolutely zero.
- Using an automated voice call instead of an SMS might be faster, but it does not help in certain cases; especially when the user is in a noisy place, or when there are frequent call drops or the transmission of the voice data itself is affected due to some technical glitch. The user, in the above circumstances, cannot hear the *OTP* very clearly, and even may miss out some of its words *or* digits.
- The application can be installed across a number of devices (Smart phones, tablets, e-book readers *etc.*,) of the user and she need not always carry her registered mobile number as in an existing telephony oriented *two-step authentication* system.
- However, linking multiple devices to the user's account might involve some risk factors, and hence the thesis prescribes to design the mobile application in such a way that
- It should be installed only on the smart phone which contains the SIM of user's registered mobile number (*MSISDN*).
- Once the user starts to use a different handset, the application installed on the former handset should not work after taking the SIM out of it; and it can no longer be used for the two-step authentication purpose. A combination of the *MSISDN* and the International Mobile Equipment Identity (*IMEI*) will be used to develop such an architecture of the mobile application. *Algorithm 9* described in [Table-13] illustrates a simple logic to prevent parallel use of the mobile application on multiple devices.

2.7 Conclusion

The thesis has proven, by dint of the mathematical analysis and an android implementation of the mobile application, that this architecture is modelled ideally for

detection of a fraud web page and the same can be used in a novel, non-telephonic approach of two-step authentication with minor modifications. Incorporating GPS and movement parameters of the user adds novelty to this approach and renders the technique the *least vulnerable* for a brute-force attack.

Table-2. Notations and symbols used in the thesis

1	E(D,K)	A symmetric cryptographic algorithm which encrypts Data(D) with a Key (K)
		Parameters: D:= Data, K:=Symmetric Key
		Corollary: $Z=\mathbf{E}(D,K) \rightarrow D=\mathbf{E}(Z,K)$
2	$E_{+Entity}(D)$ $E_{-Entity}(D)$	An asymmetric cryptographic algorithm that encrypts Data (D) with the public key (denoted by +) and private (denoted by -)of an Entity (E.g. ***Server***)
		Parameters: D:= Data
		Corollary: $W=\mathbf{E}_{+Server}(D) \rightarrow D=\mathbf{E}_{-Server}(W)$
3	$\sum(A,B,C,\ldots Z)$ *DLE*	Concatenate a number of objects (Texts or numbers) using a delimited (*DLE*)
		Parameters: Any number of texts and/or numbers

Table-1. Common approaches that steal user information

1	A compromised DNS may forward a browser pull to an unintended web site
2	Misspelled domain name might lead to a different portal with similar URL
3	Following an incorrect link from the results of searching for a web site in a search engine (E.g. Google)
4	An unofficial and unethical release of a web browser intended for collecting user data
5	Malicious browser plug-in extensions
6	Key loggers deployed by some harmful software may suppress certain key presses while a user types the web site address in the address-bar of the browser so that she visits a similarly spelled but different web site. E.g. A User may type *www.pnbindia.com* and a key logger might replace it for www.pmbindia.com which has been hosted by an attacker.
	Implementation Algorithm: *N*:= Number of Parameters *DLE*: =Delimiter *Params*[]: =Parameters Result:= EMPTY_STRING **FOR** J=0 **TO** (*N*-1) Result: = Result+ (*DLE* + *Params*[J]) **[X+Y means to concatenate Y to string X]** **NEXT** J **Output**= Result
4	QR(aText)	Create a QR code with a String (aText)
		Parameters:aText: = The text which will be the QR-coded.

5	QR^{-1} (qrImg)	Decode a QR code to extract the texts off it
		Parameters: qrImg:= A QR code image or byte stream
		Corollary: *qrImg*=**QR**(*someText*) →*someText*= $\mathbf{QR^{-1}}$ (*qrImg*)
6	$\cup_{DLE}$(S)	Tokenize a string (*S*) with a given delimiter (*DLE*)
		Parameters: S:= a String DLE:= delimiter
		Corollary: Y=∑ (A,B,C) → $\cup_{DLE}$ (Y) = {A,B,C} ***DLE***
7	$\mathbf{K_{+Entity}}$ $\mathbf{K_{-Entity}}$	The public and private keys of an entity E.g. $K_{+Server}$ means the public key of the Server whereas, $K_{-Server}$ refers to its private key

Table-3. Shared knowledgebase

1	**E**(D,K)	A symmetric cryptographic algorithm which encrypts Data(D) with a Key (K)
		Parameters: D:= Data, K:=Symmetric Key
		Corollary:Z=**E**(D,K) → D=**E**(Z,K)
2	*f*(t)	A function that operates on the timestamp, ***t.*** This is function shall be referred in the thesis as the **Timestamp Function.**
		Parameters: *t*:= *Timestamp*
3	∑ **(***A,B,C,…Z***)** ***DLE***	A function to concatenate a number of objects (Texts or numbers) using a delimited (*DLE*)
		Parameters: Any number of texts and/or numbers
		Implementation Algorithm: *N*:= Number of Parameters *DLE*: =Delimiter *Params*[]: =Parameters Result:= EMPTY_STRING **FOR** J=0 **TO** (*N*-1) Result: = Result+ (*DLE* + *Params*[J]) **[X+Y means to concatenate Y to string X]** **NEXT** J **Output**= Result
4	**DLE**	A common text delimiter
5	$\cup_{DLE}$**(***S***)**	An operation to extract (from a text) the elements delimited by a delimiter (*DLE*)
		Parameters: S:= a String DLE:= delimiter
		Corollary: Y=∑ (A,B,C) → $\cup_{DLE}$ (Y) = {A,B,C} ***DLE***
6	$\mathbf{E_{+Entity}}$ **(***D***)** $\mathbf{E_{-Entity}}$ **(***D***)**	An asymmetric encryption operations to encrypt/decrypt data (***D***) with public and private keys of an entity (***Entity***)
		Corollary: C=$E_{+Entity}$(D) → D= $E_{-Entity}$ (C)
8	**G(***lat, lng,alt***)**	This is a function that operates on the three main parameters of GPS, viz. latitude, longitude and altitude. This function is so chosen that $G(x_1, y_1, z_1) \neq G(x_2, y_2, z_2)$ for all values of x, y, z. This function shall be referred to as the location function throughout the thesis
9	**Ψ(***x,y,z***)**	This function operates on the three accelerometer parameters. This

		function is a digest function and chosen such that $M(x_1, y_1, z_1) \neq M(x_2, y_2, z_2)$ for all values of x, y, z. This function shall be referred to as the *MovementDigest* throughout the thesis.

Table-4. Algorithm 1
(One-time registration process of the mobile application)

1	The customer logs-in with her Internet banking credential through the Mobile Application
2	The server authenticates the Log-in **IF***LOG_IN_SUCESSFUL***THEN** Execute step-3 onwards **ELSE** Quit registration **END IF**
3	$\mathbf{U_1}$:= **U**(Customer's Account Information), where U is a Digest function
4	$U_1$is encrypted with the private key of the server **Outcome of this step:** $\mathbf{U_1' := E_{-Server}(U_1)}$
5	The system timestamp is fetched **Outcome of this step:** $\mathbf{T_1}$ **:= Current Timestamp**
6	Calculate the value of $\boldsymbol{f}(T_1)$ **Outcome of this step:** $\mathbf{q := f(T_1)}$
7	The public key of the server is symmetrically encrypted with ***q*** to produce a one-time key **Outcome of this step:** $\mathbf{K_S := E(K_{+Server}, q)}$
8	An application data packet containing the timestamp, the one-time key and the encrypted account digest is formed and sent to the mobile application. **Outcome of this step:** $\mathbf{T_1}$ | $\mathbf{K_S}$ | $\mathbf{U_1'}$
9	The application packet formed by the web service (i.e. server) is received by the mobile application and following computations are performed a) The fields are extracted b) $\mathbf{q:= f(T_1)}$ c) The public key of the server is obtained by decrypting the field, K_S, with the value of *q*. $\mathbf{K_{+Server} := E(K_S, q)}$ d) Finally the last field of the application field is decrypted by the public key of the server, just obtained in the previous step. $\mathbf{U_1 := E_{+Server}(U_1')}$
10	The mobile application stores the public key of the server and the digest value of the customer's account, as local application data.

Table-5. Algorithm 2
(Generation of dynamic QR codes by the web service)

1	The customer provides her Account-alias **
2	The web service receives the request over HTTPS and fetches the Account Information digest **Outcome of this step:** The web service requests a database procedure that fetches required account information of

	the Customer identified by her Account-alias. **IF***VALID_USER***THEN** ▪ Compute the Customer's Account Digest Value (U) ▪ Execute **STEP-3** onwards **ELSE** ▪ *Drop request* **END IF**
3	Generate a random number **Outcome of this step: R := rnd()**
4	The random number generated above is encrypted with the private key of the server **Outcome of this step: $R_1 := E_{-Server}(R)$**
5	The current timestamp of the system is fetched **Outcome of this step: T_S := current timestamp**
6	The value of $f(T_S)$ is calculated **Outcome of this step: q :=$f(T_S)$**
7	A one-time key is obtained by ***XOR***-ing the Account digest of the customer with the value of the timestamp function **Outcome of this step: $K_S := U \square q$**
8	R_1 is encrypted with the one-time key, K_S **Outcome of this step: $R_2 := E(R_1, K_S)$**
9	Generate a text, **Y**, by concatenating ***T_S , R_2 and R*** **Outcome of this step: $Y := \sum_{DLE}(T_S, R_2, R)$**
10	Create a QR code with the text, *Y* **Outcome of this step:qrCode:= QR(*Y*)**
11	Render this code on the web page

Table-6. *Algorithm 3*
(Verification of authenticity the web page by the mobile application)

1	The customer launches her mobile application, which is already installed on her smartphone and registered with the web service of her bank.
2	The application scans the QR code shown on the web page to extract the text information. **Outcome of this step: $Y := QR^{-1}(qrImg)$**
3	The text is parsed and un-tokenized with the delimiter, *DLE* **Outcome of this step:** $U_{DLE}(S) = \{T_S, R_2, R\}$ where all three tokens are numbers
4	Calculate the value of the timestamp function **Outcome of this step: q:= $f(T_S)$**
5	Since the mobile application had stored the customer's account digest number during one-time registration process [Table-4], the application can calculate a one-time key by XOR-ing the value of the timestamp function with the account digest number. **Outcome of this step: $K_S := q \square U$**
6	Decrypt (or, Encrypt) the value R_2 with the symmetric key K_S

	Outcome of this step: $R_1 := E(R_2, K_S)$
7	During the registration process the application had stored the public key of the server ($K_{+Server}$) So, it can perform a decryption (or, an encryption) on the value R_1 with $K_{+Server}$ **Outcome of this step:** $W := E_{+Server}(R_1)$
8	**IF** *W*=*R* **THEN** Output: Valid web page **ELSE** Output: Fraud web page **END IF**

Table-7. Algorithm 4 (Introducing GPS & accelerometer in generation of dynamic QR code by the web service)

1	The customer provides her Account-alias **
2	The web service receives the request over HTTPS and fetches the Account Information digest **Outcome of this step:** The web service requests a database procedure that fetches required account information of the Customer identified by her Account-alias. **IF** *VALID_USER* **THEN** ▪ Compute the Customer's Account Digest Value (U) ▪ Fetch customer's latest location (i.e. GPS) and movement parameters. ▪ Execute **STEP-3** onwards **ELSE** ▪ *Drop request* **END IF**
3	Generate a random number **Outcome of this step:** **P := rnd()**
4	A value ***g*** is computed using the location function using the three basic GPS parameters i.e. latitude, longitude and altitude. **Outcome of this step:** **g**:= **G** (*lat, lng, alt*); here lat, lng, lat are last updated values of latitude, longitude, altitude of the smartphone.
5	**IF** (SMARTPHONE_SUPPORTS_ACCELEROMETER) **THEN** Calculate a movement digest of the smart phone from the last updated database **M:= Ψ (F_X,F_Y,F_Z)** where {F_X, F_Y, F_Z} are the accelerometer parameters **ELSE** **M: = 0** **END IF** **Outcome of this step: M: = (**SMARTPHONE_SUPPORTS_ACCELEROMETER)**?Ψ (F_X,F_Y,F_Z) : 0**
6	**Z:= g + M**

7	Calculate a value R by *XOR*-ing*Z* and *P* **Outcome of this step: R:= Z □ P**
5	The random number generated above is encrypted with the private key of the server **Outcome of this step: R_1 := $E_{-Server}$ (R)**
6	The current timestamp of the system is fetched **Outcome of this step: T_S := current timestamp**
7	The value of ***f***(T_S) is calculated **Outcome of this step: q :=*f*(T_S)**
8	A one-time key is obtained by ***XOR***-ing the Account digest of the customer with the value of the timestamp function **Outcome of this step: K_S := U □ q**
9	R_1 is encrypted with the one-time key, K_S **Outcome of this step: R_2 := E(R_1 , K_S)**
10	Generate a text, **Y**, by concatenating ***T_S , R_2 and R*** **Outcome of this step: Y := ∑ (T_S , R_2 , R)** **DLE**
11	Create a QR code with the text, *Y* **Outcome of this step:qrCode:= QR(*Y*)**
12	Render this code on the web page

Table-8. Algorithm 5
Verification of authenticity the web page by the mobile application (modified to work with GPS & accelerometer)

1	The mobile application periodically fetches its own GPS and accelerometer parameters, stores them in local application data and send them in any predefined sequence to the bank's server. So, the application knows its last known location parameters (i.e. latitude, longitude and altitude). It calculates the location digest ***g*** **Outcome of this step: *g := G(latitude, longitude, altitude)***
2	**IF** (HAS_SUPPORTS_ACCELEROMETER) **THEN** Calculate a movement digest of the smart phone from the last updated database **M:= Ψ (**F_X ,F_Y ,F_Z**)** where {***F_X, F_Y , F_Z***} are the accelerometer parameters **ELSE** **M: = 0** **END IF** **Outcome of this step:*M: = (HAS_SUPPORTS_ACCELEROMETER)? Ψ (*F_X ,F_Y ,F_Z*) : 0***
3	**Z: = g + M**
4	The customer launches her mobile application, which is already installed on her smartphone and registered with the web service of her bank.
5	The application scans the QR code shown on the web page to extract the text information. **Outcome of this step: Y:= QR^{-1} (qrImg)**
6	The text is parsed and un-tokenized with the delimiter, *DLE*

	Outcome of this step: $U_{DLE}(S)$ = **{T_S , R_2 ,R}**where all three tokens are numbers
7	Calculate the value of the timestamp function **Outcome of this step:** **q:= *f*(T_S)**
8	Since the mobile application had stored the customer's account digest number during one-time registration process [Table-4], the application can calculate a one-time key by XOR-ing the value of the timestamp function with the account digest number. **Outcome of this step:** **K_S := q ⊕ U**
9	Decrypt (or, Encrypt) the value R_2 with the symmetric key **K_S** **Outcome of this step:** **R_1 := E (R_2 , K_S)**
10	During the registration process the application had stored the public key of the server ($K_{+Server}$) So, it can perform a decryption (or, an encryption) on the value R_1 with $K_{+Server}$ **Outcome of this step:** **W:= $E_{+Server}$ (R_1)**
11	Calculate a value *H* such that **H:= W ⊕ Z**
12	**IF***H*=*R***THEN** Output: Valid web page **ELSE** Output: Fraud web page **END IF**

Table-9. A sample definition of the timestamp function *f*(t_S)

Input	T_S := the timestamp (8-byte-long)
STEPS	1. Divide the 8 bytes in four group with 2 bytes per group 2. Calculate a checksum (C_1) on these groups of 16-bits 3. Take the even bytes and calculate their sum (S_1) 4. Take the odd bytes and calculate their sum (S_2) 5. S:= S_1 + S_2 6. S:= S *C 7. Create a byte ***B*** with the ***LSB*** (*Least Significant Bits*) 8. S: = S+B **9. Output: = S**
EXAMPLE	• Input timestamp := **Tue Jan 28 14:26:27 IST 2014** • Timestamp Value (64 bit) := **1390899387828** • Bytes obtained with this field are 0, 0, 1, 67,-40, 16, 93,-76. • So, S_1 + S_2 = 61 • The four groups of bytes are : 0000000000000000 0000000101000011 **Check sum**= $(0100)_2$ =4 1101100100010000 0101110110110100 Therefore, the value of **S** = S*C = 61*4 = ***244*** The value of the byte B = 00111010 =***58*** Hence the value of **the Output = 244 + 58 = 302**

Table-10. Definition & constraints of an ideal location function G(latitude, longitude, altitude)	
Input	1. Latitude 2. Longitude 3. Altitude
Constraints	Since the location function aims to provide a unique value for a given set of the three GPS parameters, the focus will be more on the least significant digits after the decimal point. Since, the values of the ***GPS*** parameters have 12 digits following the decimal point, in order to make the function provide considerably different values at two very near-by locations, the least significant digits are given more weightage. The constraints that must be fulfilled by an ideal Location Function are: i. $\mathbf{G}(X_1, Y_1, Z_1) \neq \mathbf{G}(X_2, Y_2, Z_2)$ if, at least, any of the three parameters differ. ii. $G(X_1, Y_1, Z_1) \neq G(X_1 + \Delta x, Y_1, Z_1)$ where Δx is a very small variation in the latitude
EXAMPLE	Assume a location has the ***GPS*** parameters {**22.37548465857538 , 88.48721790857703, 18.08639473930**} 1. We take the integral parts of the parameters in $\boldsymbol{I_1}$, $\boldsymbol{I_2}$ and $\boldsymbol{I_3}$ 2. We assign the a weightage of 1 to latitude, 2 to longitude and 3 to altitude (W_1 , W_2 and W_3) 3. We calculate a value $\boldsymbol{I = \sum W_i * I_i}$ 4. We take last weighted sum of 3 digits (after the decimal) of the parameters (**$L = 538*W_1 + 703*W_2 + 930*W_3$**) 5. We calculate weighted sum of these parameters **($F = 375*W_1 + 487*W_2 + 086*W_3$)** 6. Concatenate the integral parts of the GPS parameters and covert it to an integer **(S= 228818)** 7. **P**: = $\frac{L+F}{I}$ **= 25 (Discarding The Fractional Part)** 8. Finally concatenate S and P and convert it to integer (**Output**: **22881825**)

Table-11. Definition and constraints of an ideal *motion function* Ψ (F_X , F_Y, F_Z)	
Input	4. F_X (acceleration of device along its **X**-axis) 5. F_Y (acceleration of device along its **Y**-axis) 6. F_Z (acceleration of device along its **Z**-axis)
Constraints	$\boldsymbol{\Psi (F_{X1}, F_{Y1}, F_{Z1}) \neq \Psi (F_{X2}, F_{Y2}, F_{Z2})}$ for any two sets of 3-D motion parameters
EXAMPLE	Suppose a device is moving with the parameters {0.020030300443, 9.9453838484, -1.3394484373} We assign the weights 1, 2 and 3 to these parameters • The weighted sum of the integral parts are calculated I = 0+9-2=7 • The sum of the digits after the decimal are calculated for each of the three parameters (P_i) • **$S := \sum W_i * P_i = 255$** • Weighted sum of the last 3 digits of each of the Parameters (D_i) is calculated $L := \sum W_i * D_i =$ **2530**

	• Finally ,we calculate the **OUTPUT**:= $\mathbf{I}^3 + \frac{L}{S} = 343 + 9 = \mathbf{352}$

Table-12. Algorithm 6 (One-time Registration Process of the Mobile Application)	
1	The customer logs-in with her login credential through the Mobile Application
2	The server authenticates the Log-in **IF***LOG_IN_SUCESSFUL***THEN** Execute step-3 onwards **ELSE** Quit registration process **END IF**
3	$\mathbf{U_1}$:= **U**(Customer's Account Information), where *U* is a Digest function
4	$U_1$is encrypted with the private key of the server **Outcome of this step:** $\mathbf{U_1}' := \mathbf{E}_{\text{-Server}}(U_1)$
5	The system timestamp is fetched **Outcome of this step:** $\mathbf{T_1}$ **:= Current Timestamp**
6	Calculate the value of $f(T_1)$ **Outcome of this step:** $\mathbf{q} := f(T_1)$
7	The public key of the server is symmetrically encrypted with ***q*** to produce a one-time key **Outcome of this step:** $\mathbf{K_S} := \mathbf{E}(K_{\text{+Server}}, q)$
8	An application data packet containing the timestamp, the one-time key and the encrypted account digest is formed and sent to the mobile application. **Outcome of this step:** $\mathbf{T_1}$ | $\mathbf{K_S}$ | $\mathbf{U_1}'$
9	The application packet formed by the web service (i.e. server) is received by the mobile application and following computations are performed a) The fields are extracted b) $\mathbf{q} := f(\mathbf{T_1})$ c) The public key of the server is obtained by decrypting the field, K_S, with the value of ***q***. $\mathbf{K}_{\text{+Server}} := \mathbf{E}(\mathbf{K_S}, \mathbf{q})$ d) Finally the last field of the application field is decrypted by the public key of the server, just obtained in the previous step. This yields the account digest. $\mathbf{U_1} := \mathbf{E}_{\text{+Server}}(\mathbf{U_1}')$
10	The mobile application stores the public key of the server and the digest value of the customer's account, as local application data.

Table 12-A. Algorithm 7 (One-time QR-challenge generated by the web service)	
1	The user attempts a log-in with her login credentials. The web services does not readily allow the login; instead holds the login until the user responds to the One-time challenge it generates.

2	The web service receives the request over HTTPS and fetches the Account Information digest **Outcome of this step:** The web service requests a database procedure that fetches required account information of the Customer identified by her Account-alias. **IF***VALID_USER***THEN** ▪ Compute the Customer's Account Digest Value (***U***) ▪ Fetch customer's latest location (i.e. ***GPS***) and movement parameters. ▪ Suspend the log-in process till it generates the one-time challenge (**STEP-3** onwards), displays it and the user responds to it. **ELSE** ▪ *Drop the log-in process* **END IF**
3	Generate a random number and stores it in a short-time session. (against the user-id that attempted login) **Outcome of this step: P := rnd()**
4	A value ***g*** is computed using the location function using the three basic GPS parameters i.e. latitude, longitude and altitude, of the last known (updated) location of the user. **Outcome of this step: g**:= **G** (latitude, longitude, altitude);
5	**IF** (SMARTPHONE_SUPPORTS_ACCELEROMETER) **THEN** Calculate a movement digest of the smart phone from the last updated database **M:= Ψ (F_X,F_Y,F_Z)** where {F_X, F_Y, F_Z} are the accelerometer parameters **ELSE** **M: = 0** **END IF** **Outcome of this step: M: =** **(**SMARTPHONE_SUPPORTS_ACCELEROMETER)**?Ψ (F_X,F_Y,F_Z) : 0**
6	**Z:= g + M**
7	Calculate a value R by *XOR*-ing*Z* and *P* **Outcome of this step: R:= Z □ P**
5	The random number generated above is encrypted with the private key of the server **Outcome of this step: R_1 := $E_{-Server}$ (R)**
6	The current timestamp of the system is fetched **Outcome of this step: T_S := current timestamp**
7	The value of ***f***(T_S) is calculated **Outcome of this step: q :=*f*(T_S)**
8	A one-time key is obtained by ***XOR***-ing the Account digest of the customer with the value of the timestamp function **Outcome of this step: K_S := U □ q**
9	R_1 is encrypted with the one-time key, K_S **Outcome of this step: R_2 := E(*R_1* , *K_S*)**
10	Generate a text, **Y**, by concatenating ***T_S* , *R_2* and *R***

	Outcome of this step: $\mathbf{Y := \sum (T_S , R_2)}$ **DLE**
11	Create a QR code with the text, ***Y*** **Outcome of this step:qrCode:= QR(*Y*)**
12	Render this QR-code on the web page. This is the one-time challenge which the server sends without using automated voice calls or SMS.

Table 12-B. Algorithm 8.
Process of generating the response to the One-time challenge by the mobile application

1	The mobile application periodically fetches its own GPS and accelerometer parameters, stores them in local application data and send them in any predefined sequence to the bank's server. So, the application knows its last known location parameters (i.e. latitude, longitude and altitude). It calculates the location digest ***g*** **Outcome of this step:** ***g := G(latitude, longitude, altitude)***
2	**IF** (HAS_SUPPORTS_ACCELEROMETER) **THEN** Calculate a movement digest of the smart phone from the last updated database **M:= Ψ (**F_X ,F_Y ,F_Z**)** where {$\boldsymbol{F_X}$**,** $\boldsymbol{F_Y}$ **,** $\boldsymbol{F_Z}$} are the accelerometer parameters **ELSE** **M: = 0** **END IF** **Outcome of this step:*M: = (HAS_SUPPORTS_ACCELEROMETER)? Ψ (***F_X ,F_Y ,F_Z***) : 0***
3	**Z: = g + M**
4	After filling-in her login credentials to the web site, the user launches the mobile application on getting the One-time challenge on the browser's screen
5	The application scans the QR code shown on the web page to extract the text information. **Outcome of this step:** $\mathbf{Y:= QR^{-1} (qrImg)}$
6	The text is parsed and un-tokenized with the delimiter, *DLE* **Outcome of this step:** $\mathbf{Ʉ_{DLE}}$(S) = **{T_S , R_2 ,R}**where all three tokens are numbers
7	Calculate the value of the timestamp function **Outcome of this step:** **q:= *f*(**T_S**)**
8	Since the mobile application had stored the customer's account digest number during one-time registration process [*Algorithm-6*], the application can calculate the one-time key by XOR-*ing* the value of the timestamp function with the account digest number. **Outcome of this step:** $\mathbf{K_S := q \square U}$
9	Decrypt (or, Encrypt) the value R_2 with the symmetric key $\mathbf{K_S}$ **Outcome of this step:** $\mathbf{R_1 := E (R_2 , K_S)}$
10	During the registration process the application had stored the public key of the server ($K_{+Server}$) So, it can perform a decryption (or, an encryption) on the value R_1 with $K_{+Server}$ **Outcome of this step:** $\mathbf{W:= E_{+Server} (R_1)}$
11	Calculate a value ***H*** such that **H:= W □ Z**

	Outcome of this step: The response to the one-time challenge*i.e. **H***
12	The user submits this response (**H**) to the web server which check the following **IF***H*=*P***THEN** The user is considered to have got through this *2-stepverification.* [Ref. **Step-3*Algorithm 7***] **ELSE** **IF** (*NO_OF_ATTEMPTS > MAX_NO_OF_INCORRECT_REPONSE*) **THEN** *Blocks log-in access of the account temporarily* **ELSE** o Repeat ***Step-3*** of **Algorithm 7** o **NO_OF_ATTEMPTS**:=**NO_OF_ATTEMPTS** *+1* **END IF** **END IF**

Table-13. *Algorithm 9*
Prevention of multiple devices using the authenticating App

1	The application is installed on *Device –X*
2	M:= the registered mobile number (MSISDN) of the user
3	The user launches the App.
4	The Application checks for all the Telephony Units within the smart phone. A telephony unit refers to an active *SIM* slot in the device. **Outcome of this step: T[]:** = The available telephony units **(N.B. there is commonly one telephony unit in a device; but in case of dual SIM phones, it is two. Rarely there are more than 2 telephony units in a single smart phone till date)**
5	The application checks whether the registered mobile number (*MSISDN*) is inside any of the telephony units. If the MSISDN is not attached to any of the telephony units, the application blocks itself. *FLAG***:=** *false* **FOR***I*=0 **TO** (*T*.***length***-1) **STEP***1* **IF (T[I]** *is attached to MSISDN* **M) THEN** *FLAG:= true* **BREAK;** **END IF** **NEXT** I **IF** (FLAG== *false*) **THEN** **Block_application()** **ELSE** **Properly_Work()** **END IF**

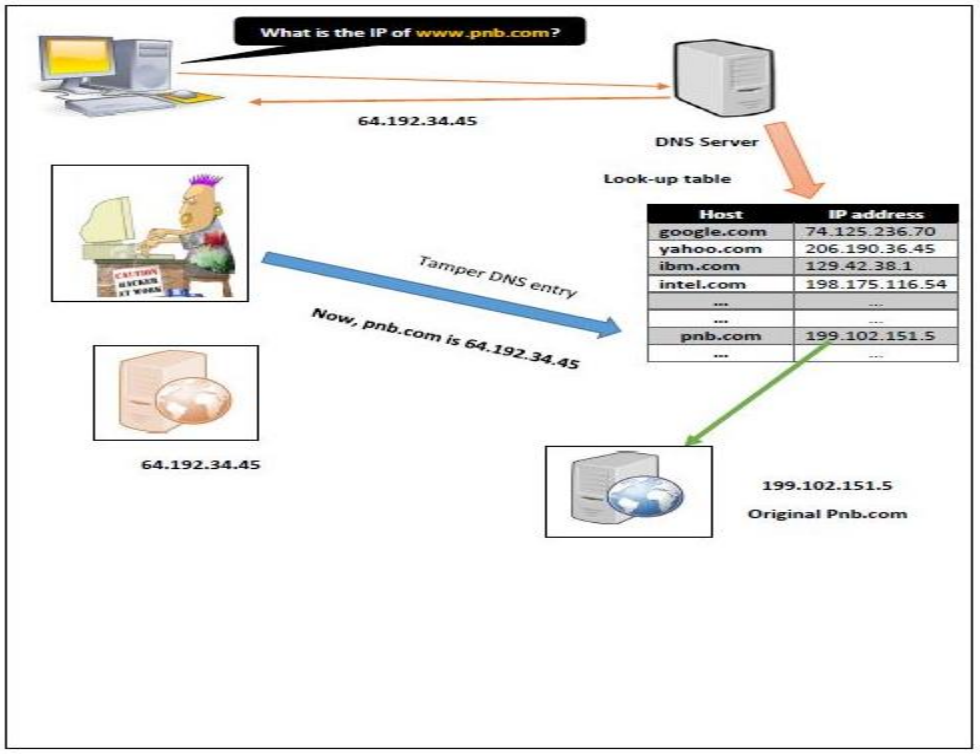


Figure 1: A COMPROMISED DNS

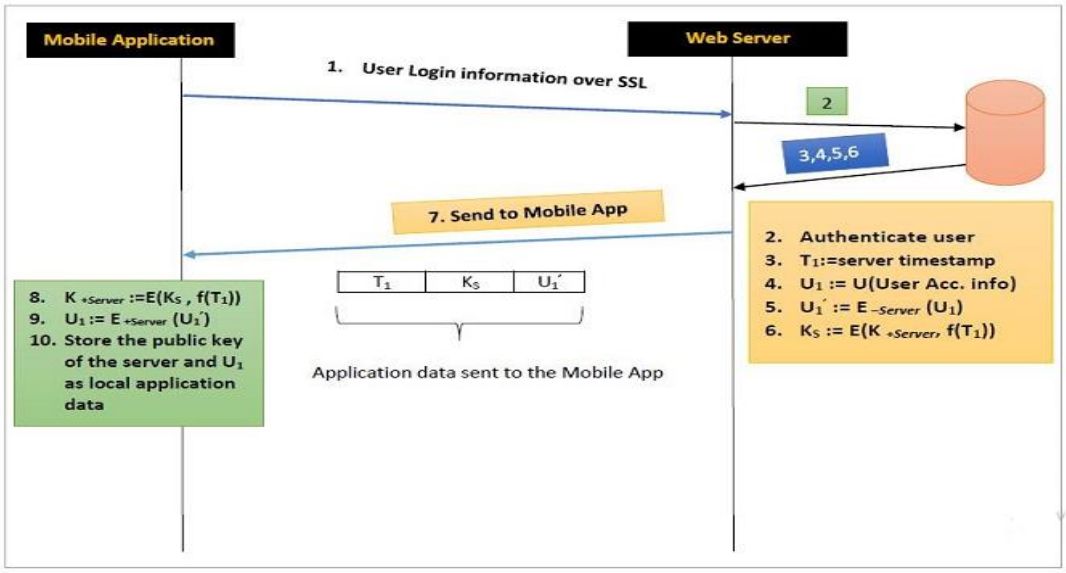
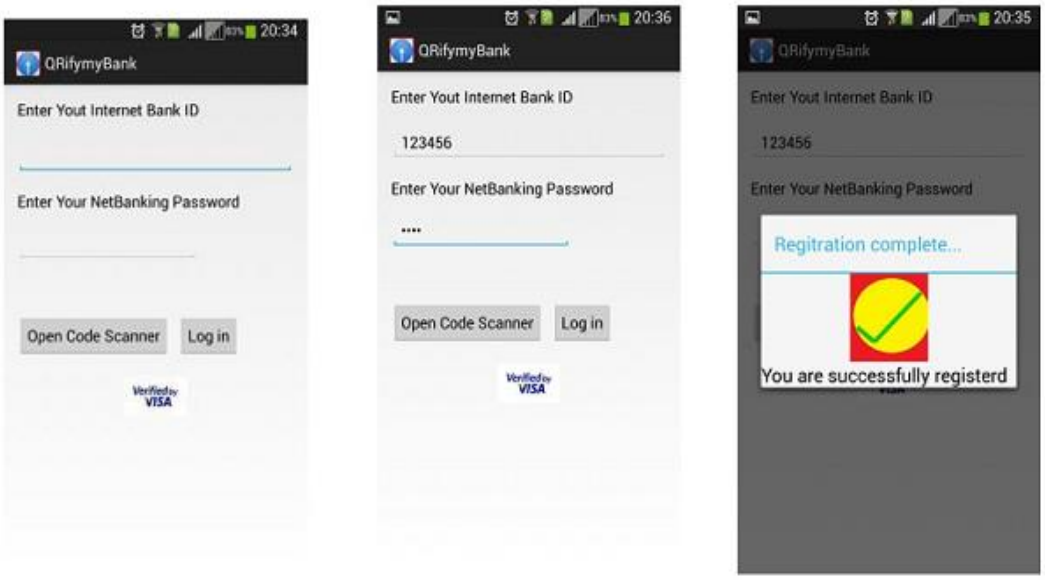


Figure-2:

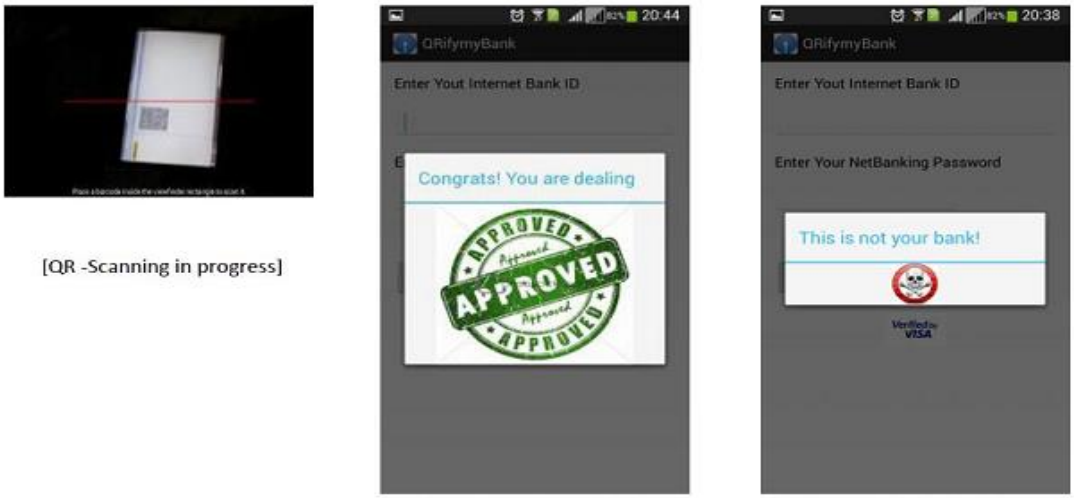
Timeline diagram showing one-time set up of the mobile application



[Application launch screen] [Registration Process] [Registration Completed]

FIGURE-3

Screen shots of the mobile application



[QR -Scanning in progress]

[A passed-verification] **[A failed verification]**

FIGURE-4

MOBILE APPLICATION SCREEN SHOTS OF DETECTION OF A FRAUD WEB PAGE

[QR Code on the portal containing encrypted OTP]

[Scanning the QR code by the mobile Application the OTP is decrypted]

FIGURE-6: Screen shots of two step authentication accomplished by encrypted QR code and Mobile App

Acknowledgements

The author [1] gratefully acknowledges kind contributions of Prof. Debesh Choudhury, Ex-Scientist, ISRO, Dr. ProbalSengupta, Scientist & Director, Alumnus Software Limited, Mr. Srijeeb Roy, Senor Java and Mobility Evangelist, Tata Consultancy Services, whose affectionate and constant support and inspiration has been the cue in his [1] researches.

References

[1] N. A. Hamidi, G. K. M. Rahimi, A. Nafarieh, A. Hamidi and B. Robertson, "Personalized Security Approaches in E-banking Employing Flask Architecture over Cloud Environment", Procedia Computer Science, vol. 21, **(2013)**, pp. 18-24, ISSN 1877-0509, http://dx.doi.org/10.1016/j.procs.2013.09.005.
[2] S. Furnell, "E-commerce security: a question of trust, Computer Fraud & Security", vol. 2004, Issue 10, **(2004)** October, pp. 10-14, ISSN 1361-3723, http://dx.doi.org/10.1016/S1361-3723(04)00122-8, (http://www.sciencedirect.com/science/article/pii/S1361372304001228)
[3] P. Mukherjee and B. J. Jansen, "Performance analysis of keyword advertising campaign using gender-brand effect of search queries", Electronic Commerce Research and Applications, Available online, **(2014)** January 2, ISSN 1567-4223, http://dx.doi.org/10.1016/j.elerap.2014.01.001.
[4] G. Oosthuizen, "Security issues related to E-commerce", Network Security, vol. 1998, Issue 5, **(1998)** May, pp. 10-11, ISSN 1353-4858, http://dx.doi.org/10.1016/S1353-4858(98)80120-7.
[5] H. Suryotrisongko and B. S. Sugiharsono, "A Novel Mobile Payment Scheme based on Secure Quick Response Payment with Minimal Infrastructure for Cooperative Enterprise in Developing Countries", Procedia - Social and Behavioral Sciences, vol. 65, **(2012)** December 3, pp. 906-912, ISSN 1877-0428, http://dx.doi.org/10.1016/j.sbspro.2012.11.218.
[6] S. Sengupta, "An approach to provide a network layer security model with QR code generated with shuffled GPS parameters as embedded keys traveling over Internet using existing IPv4 mechanism, Computer Networks, vol. 57, Issue 11, **(2013)** August 5, pp. 2313-2330, ISSN 1389-1286, http://dx.doi.org/10.1016/j.comnet.2013.04.003.
[7] F. Belanger, J. S. Hiller and W. J. Smith, "Trustworthiness in electronic commerce: the role of privacy, security, and site attributes", The Journal of Strategic Information Systems, vol. 11, Issues 3–4, **(2002)** December, pp. 245-270, ISSN 0963-8687, http://dx.doi.org/10.1016/S0963-8687(02)00018-5.
[8] B. Yu, Z. Guo-Sun and Z. Liang, "Additional service security of e-commerce in mine enterprises", Procedia Earth and Planetary Science, vol. 1, Issue 1, **(2009)** September, pp. 1574-1580, ISSN 1878-5220, http://dx.doi.org/10.1016/j.proeps.2009.09.242.
[9] M. Tomlinson, "Tackling E-commerce Security Issues Head On, Computer Fraud & Security, vol. 2000, Issue 11, **(2000)** November 1, pp. 10-13, ISSN 1361-3723, http://dx.doi.org/10.1016/S1361-3723(00)11017-6.

Authors

Soham Sengupta, he holds an M.Tech in the field of mobile computing and B.Tech (*Hons*. In Information Technology) He has an excellent academic and professional career till date. He has been in the field of teaching and research for last ten years. He has a strong hold of the impelling technologies and is held in high esteem in the Industries. His research interest covers Java and other open sources, Object Oriented Design patterns, Mobile Computing and communication, Bluetooth Applications and JSR-82, Computer Networks (Protocols, Security, Augmentation and Applications). He is an *AndroidExponent*, also known for his proficiency in Java, RIA frameworks, Cross Platform Mobile Apps development and giving innovative cost effective solutions. Augmented Reality and Computer vision are his present passions. By profession, he is serving the department of Information Technology as an Assistant Professor at JIS College of Engineering, Kalyani, India; and he consulted for different industries, like ZreyasTechnolohy Inc. as an Android Architect and Device Integration Expert. Founder CTO to TECH IT easy Labs, he has a vast experience in the Industry like IBM, TCS, Yotto Labs and Touchstone Tie up Private Limited etc. His passions include relating inter-disciplinary topics, and strongly denies a barrier of subjects or topics. Some of his contributions to open sources can be found at http://sourceforge.net/users/sohamsengupta, Google Android market and http://sohamsironline.weebly.com.

Debasree Chanda Sarkar, he was felicitated with a Ph.D in engineering from Jadavpur University in the year 2005. She has obtained her M.E from Bengal Engineering and Science University, Shibpur in the year 1994.She earned her B.E degree in Electronics and Telecommunication Engineering from Bengal Engineering College (Presently known as Bengal Engineering and Science University, Shibpur) in the year 1991. She is presently working as Scientific Officer (Associate Professor Rank) at the Dept. of Engineering & Technological Studies, University of Kalyani. Her area of research includes, Microstrip Antenna, Microstrip Filter, Frequency Selective Surfaces, and Artificial Neural Network. She has contributed to numerous research articles in various journals and conferences of repute.

S. Biswas, she was felicitated with a Ph. D in engineering from Jadavpur University in the year 2004. He has obtained his M.E from Jadavpur University in the year 1995. He earned his B.E degree in Electronics and Telecommunication Engineering from Bengal Engineering College (Presently known as Bengal Engineering and Science University, Shibpur) in the year 1990. He is presently working as Associate Professor at the Dept. of Engineering & Technological Studies, University of Kalyani. His area of research includes, Microstrip Antenna, Microstrip Filter, Frequency Selective Surfaces, and Artificial Neural Network. He has contributed to numerous research articles in various journals and conferences of repute

ParthaPratim Sarkar, he was felicitated with a Ph.D in engineering from Jadavpur University in the year 2002. He has obtained his M.E from Jadavpur University in the year 1994. He earned his B.E degree in Electronics and Telecommunication Engineering from Bengal Engineering College (Presently known as Bengal Engineering and Science University, Shibpur) in the year 1991. He is presently working as Professor at the Dept. Of Engineering & Technological Studies, University of Kalyani, His area of research includes, Microstrip Antenna, Microstrip Filter, Frequency Selective Surfaces, and Artificial Neural Network. He has contributed to numerous research articles in various journals and conferences of repute. He is also a life Fellow of IETE and IE (India).