Mobix: A Software Proposal Based Authentication Service for Mobile Devices

Ali Alsaadi, Anup Kumar, Bin Xie, Adel Elmaghraby
Department of Computer Engineering and Computer Science
University of Louisville
Louisville, USA
ali.alsaadi,anup.kumar,adel@louisville.edu

Abstract—In this paper, MobiX technology provides security for online services using mobile devices with a wide range of customers including governments, military, business enterprises, and scientific organizations, and individuals have been proposed. MobiX is a software security solution technology to enhance the networked mobile device trustworthiness, application integrity, preventing users from malware attacks. MobiX provides two authentication layers and two additional security layers in addition to the conventional username-password for mobile devices in a unique system design. MobiX is fully implemented by software; therefore, it can be quickly deployed into different legacy mobile devices. More importantly, the novel device authenticity is based on the one-way a keyless cryptographic hash function and on the Generalized Concatenated Code Physically Unclonable Function (GCCPUF) to effectively protect the users from different attack vectors such as insiders, spoofing attack, or identity theft attacks. On the other hand, the app authenticity prevents malware and app-tempering cybersecurity attacks via an efficient and secure remote attestation protocol. The comprehensive protection architecture of MobiX would enable M-commerce with superior security capabilities compared with the existing solutions.

Keywords— threats; spoofing attack; PUF, keyless hash function; Generalized Concatenated Code; M-commerce

I. INTRODUCTION

Nowadays, internet is used everywhere - everyone has multiple devices. These online services via the Internet have a number of both advantages and disadvantages. The benefits for organizations, providing such services include: (1) low cost of operation of online systems; (2) ability to integrate with the client's accounting systems; (3) the availability to provide offer Internet services to the end user; (4) customer Loyalty benefits for active uses. The disadvantages are, first, weakly protected Internet solutions from unauthorized access. Maintaining a level of protection at an appropriate level requires considerable costs, which can be afforded larger banks. Currently, hardware-based techniques can provide strong security; however, these approaches require additional hardware components (e.g., Trusted Platform Module (TPM)), which are costly, and importantly, these modules cannot be integrated into billion of existing mobile devices in the market. On the other hand, software only approaches provide very weak security that are not suitable to online services with sensitive information. Several solutions [1–4] have been developed to enhance the username-password security by adding an extra layer of protection. For example, Bank of America uses SafePass [4] to secure its online banking transactions, where the user uses a 6-digit passcode sent via the text SMS messages to the registered mobile device for authentication. However, the two-factor authentication approaches have been recently compromised by the use of Trojans [5]. The other recent advanced methods, e.g., using biometrics such as fingerprint, have been shown to be vulnerable to cybersecurity attacks, e.g., [6,7]. Different from the existing solutions, Hybrid Network - Software Security Solution (MobiX) technology is proposed to enhance the networked mobile device trustworthiness, application integrity, and user authority preventing users from malware attacks. MobiX is fully implemented by software; therefore, it can be quickly deployed into different legacy mobile devices. The rest of this paper is structured as follows: we introduce the technical innovation of MobiX system and software Architecture in section 2. In section 3, we describe our new design and implement of MobiX system. Conclusions are discussed in security analysis is earned out in section 4.

II. THE INNOVATION

Hybrid Network-Software System Architecture

The proposed MobiX system encompasses several novel technologies that enable device trustworthiness, new application of mutual authentication, application integrity, and user authority preventing users from malware attacks. Different from the existing hardware or software-based approaches which are either impractical or vulnerable to sophisticated cybersecurity attacks, we propose a novel network-based system architecture to provide strong security for M-commerce services. Specifically, MobiX provides two authentication layers containing additional security sub-layers. These are in addition to the conventional username-password sub-layers. These are in addition to the conventional username-password sub-layers. These are in addition to the conventional username-password sub-layers. These are in addition to the conventional username-password sub-layers.
a device-based fingerprint validation algorithm to verify the device authenticity. On the other hand, a user can register multiple devices for an application. Device authenticity protects users from different attack vectors such as insiders, spoofing attack, or identity theft attacks.

b) Mutual Authentication: is a security feature of MobiX technology that a well-designed solution to protect against of online fraud such as man in the middle attacks, Trojan horses, shoulder surfing, and pharming. Mutual authentication is used in conjunction with the device authenticity for optimum security.

2) The Second Authentication Layer(Application Authenticity): Application authenticity is another key security feature of MobiX technology that prevents malware stealing information in the mobile device. MobiX validates the application authenticity from three aspects:

a) Code Integrity: Application code is validated to prevent forge software. Only authentic program code issued by the authority can be executed at the smartphone.

b) Execution Integrity: It prevents malware code injection during the application execution. It is able to detect and protect the software from tampers, such as code modification and memory copy attacks.

c) Data Integrity: Data integrity validates the data generated or transmitted from the application is correct without maliciously modification.

Figure 1 illustrates a simple scenario that shows MobiX security operations preventing unauthorized access. In this scenario, an authentic user, Alice, tries to access the services (e.g., purchasing or banking transactions) by using her registered credentials and device (e.g., a smartphone). As shown in Figure 1, the service authentication is conducted in four steps. In the first step, the service request is sent to the Service Authority for authentication. Upon receiving the request, the Service Authority will parse the request and sent the second authentication request to the service for authentication. Upon receiving the request, the service will verify and forward it to the proposed MobiX for security verifications. MobiX then sends a request for authentication to the user/device. MobiX conducts two layers of Authentication, for each layer two-factor authentication, including the user’s credentials, device, and application to determine if the requested service can be granted.

III. DESIGN AND IMPLEMENT MOBIX SYSTEM

The MobiX System procedure is illustrated in Figure 2. The proposed MobiX system encompasses several novel technologies that enable device trustworthiness, new application of mutual authentication, application integrity, and user authority preventing users from malware attacks. Different from the existing hardware or software-based approaches which either impractical or vulnerable to sophisticated cybersecurity attacks, we propose a novel network-based system architecture to provide strong security for M-commerce services. Specifically, MobiX provides two authentication layers which containing of layers three additional security layers in addition to the conventional username-password on the mobile applications:

A. PUF-based Registration

This is the initial step where the user registers to the Authority with the help of the mobile device. If the registration is successful, the server obtains and stores a record of the user’s profile in its database that will be used for future authentication. The PUF-based registration protocol is illustrated in Figure 3 including the following steps:

1. Selecting operation group and challenge: The MobiX authority S sends challenge c to the MobiX Agent U along with the description of a group G, denoted by <G_q> (q is the group order, i.e., a prime number) and the group generator g. G_q could be a subgroup of the multiplicative group \( \mathbb{Z}_p^* \) for a
prime $p$. The information is encrypted by the Authority’s private key, $\langle c \| g, g \rangle$.

2. Incorporating user credentials: $U$ decrypts and incorporates received information with the user credentials (i.e., username/password). This is analogous to asking the user to enter his/her desired password. The incorporated information is encrypted by a hash function $H(c \| \text{pwd}), \langle G_r \rangle, g$, where pwd denotes the user passcode, and sent to the mobile device $D$.

3. Determining local challenge and response: The device then computes its own challenge based on the received information as $d = H(H(c \| \text{pwd}), \langle G_r \rangle, g)$. Device $D$ generate

4. $U$ encrypts and forwards $\langle g \| p \rangle$ to the MobiX Authority $S$, which stores the information along with $c, g, \langle G_r \rangle$.

- Accessing without Device: This is an option using which the user can access to the system without using the device. In order to achieve this, the user needs to answer additional challenge questions. The answers are stored at the Authority.

B. Mutual Authentication-based Keyless Hash Function Protocol

The mutual authentication procedure is illustrated in Figure 4 including the following steps:

- Sending first message: The mobile device sends the first message of a request and the response $h_R$ to Authority server, it is known only to advance the legitimate correspondents request of the Authority server and the second respondent's answer $h_S$ of the mobile device. Wherein the request is generated by calculating the mobile device keyless hash function (e.g., algorithms [8-10]) argument, constitutes a calculation result $h_r = h(b_r \| Z) = h(M_r \| h_i(M_r \| Z))$ in the same hash function $h(v)$ random challenge and the shared secret $h_s = h(M_r \| Z)$, concatenated with the value of a random number, to which is added a random number itself.

  Decrypt and verification: Once received server authority decrypts and calculating the result of $h_R$ and $h_s$ by using a keyless hash function $\tilde{h}_s$ and $\tilde{h}_r$ (i.e., $\tilde{h}_s = h(M_r \| Z)$, $\tilde{h}_r = h(b_r \| Z) = h(M_r \| h_i(M_r \| Z)$), then verify if $\tilde{h}_r = \tilde{h}_s$, then send the second message $\tilde{h}_r$ to mobile device.

  Sending second message: The second message (i.e. response message) sends from Authority server to mobile device which includes values respondent keyless hash function $\tilde{h}_r$ and the random challenge. The mobile device receives the reply, conducts the keyless hash function of the request and compares the results with the received message.

  Verification: In case of exact correspondence between the transformations (i.e. $\tilde{h}_i = \tilde{h}_r$), the protocol is successfully terminated.

In this exchange of information Figure 5 by reducing the amount of transmitted messages reduced the time required to perform mutual authentication of correspondents. Persistence method determined by the selection algorithm [11] calculating unidirectional keyless hash function and does not require a periodic change of the shared secret, which makes it long.
C. GCC-PUF Cryptographic Key Generation G3CKG

We now describe a scheme to reliably generate unclonable cryptographic key based on the PUF circuits of the mobile devices. A PUF allows to generate the unique key “on-the-fly” by using intrinsic physical properties of the PUFs [12,13]. Unclonable means the hardness of manufacturing two PUFs with the same challenge-response-behavior. There are many possibilities to realize PUFs, e.g. delay-based (e.g. Ring Oscillator PUFs) or memory-based (e.g. SRAM PUFs). PUFs can be used in order to realize secure key generation and storage for cryptographic applications. Due to static randomness over the PUFs lifetime, it is possible to regenerate a key repeatedly on demand instead of storing it permanently. As described above, PUF responses are not exactly reproducible and therefore a response cannot be used as key directly. To address these issues, the proposed G3CKG provides a novel method utilizing the power-up states of Code Construction (CC) theory to generate security keys [14]. We start with the description of GCC-PUF key generation.

GCC-PUF Key Generation in Actual Mobile Devices: Code-Offset Construction [15] is used to realize key reproduction, the system consists of enrollment and reproduction phases as illustrated in Figure 5.

Enrollment Phase: The enrollment phase consists of the following: 1) the Authority server generate response \( r \) from a given challenge \( c \) and sends to the Helper Data Generation (HDG) at mobile device (i.e. \( r = c_w + e \), \( e \) is error), 2) Design a code \( C \) for key reproduction with a block error probability smaller than a certain threshold (described in the next subsection), 3) subtracts a random codeword \( c_w \) of a given code \( C \) from \( r \) and stores the result \( e = r - c_w \) in the Helper Data Storage (HDS), 4) The response \( r \) deleted, therefore, if an attacker is able to read this storage, he is left with an uncertainty as large as the number of codewords.

Reproduction Phase: The reproduction phase consists of the following: 1) the Authority server send response \( r' \) from a same challenge \( c \) to the mobile device which is likely to differ a little from \( r \) (i.e. \( r' = r + e' = c_w + e + e' \)), resulting from a Binary Symmetric Channel (BSC) with crossover probability \( p \), where \( p \) is given by the PUF), 2) the Key Reproduction (KR) process is able to reproduce the first time response \( r \) by decoding \( r' - e = c_w + e + e' - e = c_w + e' \) (i.e. \( \tau = c_w + e' \)) with a decoder of the code \( C \), 3) decoding result hashed by a cryptographic hash function to obtain a uniformly distributed cryptographic key.

D. Secure Remote Code Integrity Attestation (SRCIA) Protocol

In this subsection, we will describe a GCC-PUF-based remote attestation protocol between the mobile device...
(i.e., prover) and the authority server (i.e., verifier) with the aid of the user. The objective of attestation is to convince the verifier that the device is in a trusted software state \( T \). Different from the existing software-based attestation schemes, the proposed scheme does not assume a secure communication link between \( D \) and \( S \). More importantly, the proposed scheme provides the capabilities of hardware attestation, i.e., the operation is running on the designated device, defeating impersonation/man-in-the-middle attacks.

The proposed protocol is illustrated in Fig. 7 consisting of the following steps:

- **Attestation Request**: The local MobiX issues an attestation request for a code \( C \), targeting to the MobiX authority. As illustrated in Figure 6, the request includes the user ID and credentials. Before sending to \( U \), the information is encrypted by a keyless hash function with the nonce \( Z \), \( h_1 = h(\text{User ID} \| \text{Credentials} \| h \| Z) \). The nonce \( Z \) is to avoid replay attacks such as Denial-of-Service attacks (DoS) or Distributed Denial-of-Service attacks (DDoS).

- **Response and Group Description**: Once received the authentication and attestation request, the server decrypts and first lookups in its database for the ID and associated credentials. Once found, after that calculating the result of \( h_R \) and \( h_1 \) by using a keyless hash function \( \hat{h} \). If valid, the server retrieves a response \( R \) of a challenge \( c \), its associated information, i.e., operation group \( g \), group generator \( g \), and the syndrome \( P \). The nonce \( Z \) is to ensure the request is not replayed. The server concatenates the received information and encrypts the resultant information \( \hat{h} = \hat{h}(\text{User ID} \| \text{Credentials} \| P \| \langle Gq \rangle \| g \| Z) \) to the local MobiX agent \( U \), and rejects, otherwise.

- **User Consent**: Upon receiving the information sent from the server, the local MobiX agent decrypts and verify if \( \hat{h} = \hat{h} \) then requests for a consent from the user. Once agree, the user enters the passcode and the information is forwarded to the device \( D \), and rejects, otherwise.

- **Compute Challenge and Checksum**: Once received the information, the device \( D \):

1. Calculates the response: This procedure is carried out in the same manner as the “Reconstruct the response” in the Authentication procedure. As a result, the device will output two parameters \( \hat{c}_w = H\left( g, g', t, N \right) \) and \( w = v - r \hat{c}_w \mod q \).

2. Calculates the code checksum: The device \( D \) evaluates the checksum which is the fingerprint of the program \( C \) as \( z = \text{cksum}(C) \).

The generated information is sent to the local MobiX agent for further processing before sending to the authority server.

- **Encrypt Response**: The local MobiX agent concatenates the received information and encrypts the resultant data using \( K_z \) as \( \langle \hat{c}_w \| z \rangle_{K_z} \). The response along with the code checksum is sent to the authority server for verification.

- **Software Code Integrity Verification**: The verification procedure is performed at MobiX authority when it receives the data message from a local MobiX agent. It is performed in two steps:

1. **User-Device Authentication**: This procedure is to validate the authenticities of the device and user. It is carried in the same manner as the “Verification” step of the Authentication procedure, i.e. if \( \hat{c}_w = H\left( g, g', r, Z \right) \), where \( r' = g \cdot r \cdot c_w \), the server move to code integrity verification.

2. **Code Integrity Verification**: The code integrity verification is straightforward. MobiX authority first extracts the checksum \( Z \) from the received message. It then compares the extracted checksum \( Z \) with its computed one \( Z' = \text{cksum}(C) \). It is worth noting that the checksum \( Z' \) is evaluated by using the program \( C \) maintained at the MobiX authority, which is not accessible by attackers. \( z' \) indicates that the code integrity at the mobile device and the remote attestation procedure is valid. MobiX authority consequently returns a positive response, \( \langle \text{Reply}(C) \| Z \rangle_{K_z} \), to the local MobiX agent. Based on the received integrity, the local MobiX agent will decide whether the code \( C \) is allowed to run or not.

### IV. Security Analysis

The proposed protocol security is analyzed based on the following facts:

- **Avoid repetition attack**: To avoid repetition attack, one-way authentication model in the earlier transmitted request provide resistance random \( Z \) to the accumulation statistics transmitted messages [16].

- **Increases the efficiency of the communication channel**: the one-way authentication model allows reducing the number of transmitted messages to two, which reduces the time of authentication for communication systems operating on different channels without compromising computing resistance method prototype. This increases the efficiency of the communication channel and reduces the time required to obtain access to information resources.

- **Zero-knowledge Response**: As the device does not send the response \( r \) in a clear form, instead it sends \( g' \) to the authority server. Therefore, the device reveal nothing about the response of the input challenge. In this way, the protocol can prevent the attackers from learning the response by observing the device.

- **The Hardness of the Discrete Algorithm**: The strong security of the proposed scheme comes from the hardness of the discrete algorithm. Particularly, there is no efficient algorithm to recover \( r \) given \( g' \) [17]. In other words, the attackers can compromise \( g' \), but they learn nothing about \( r \).

- **Untampered PUF**: Importantly, the security of the proposed algorithm is achieved based on the uncloneable characteristic of the device [12]. These capabilities combined
with additional user’s credentials information provide strong security for the proposed system.

V. CONCLUSION

In this paper, we design a secure network-based system to solve the problem of hardware-based techniques which can provide strong security; however, these approaches require additional hardware components (e.g., Trusted Platform Module (TPM)), which are costly, and importantly, these modules cannot be integrated into billion of existing mobile devices in the market. MobiX offers hardware-equivalent security capabilities by fully implemented software. It incorporates four novel technologies including protocol of keyless cryptographic hash cryptography shared secret with a random variable generation, code construction unclonable cryptography key reproduction, secure and efficient multi-factor authentication, and secure remote code integrity attestation. MobiX offers a provable and mutual service authentication based on the keyless cryptographic hash function and unclonable service authentication based on the GCCPUF-based cryptographic keys. The proposed system works well under the condition of untrusted network connection between the user and service provider. Importantly, even in the cases where the users’ credentials are compromised, MobiX can still prevent the attackers from accessing the authentic services. Additionally, MobiX has a new client-authority system and software structure enabling secure remote software, device, and identify attestations that not provided by the existing solutions.

REFERENCES