WYK: Mobile Device Authentication Using the User's Address Book

Mehari Msgna, Sokratis Katsikas, and Vasileios Gkioulos

Norwegian University of Science and Technology (NTNU),
Teknologivegen 22, 2815 Gjøvik, Norway
{mehari.g.msgna,sokratis.katsikas,vasileios.gkioulos}@ntnu.no

Abstract. Authenticating a user the correct way is paramount to IT systems, where the risk is growing more and more in number and complexity. This is specially important in mobile phones, where a number of applications require continuous device authentication following the *Point-of-Entry* user authentication. Existing common approach in systems that require strict security rules and regulations is to use a *One-Time-Password (OTP)*. Usually the OTP is generated using a special hardware device or another application that is synchronised with the back-end system. Another approach is to use SMS based activation/approval codes such as used by Telegram, Facebook, Twitter and other social media platforms. However, this approach has three major drawbacks; (1) it requires active user participation/interaction which could be annoying if repeated continuously, (2) SMS messages can be accessed by service provider's employees, and (3) it does not consider the authenticity of the device from which the services are being accessed. The later is particularly serious as access sessions can be hijacked by malicious entities. In this paper, we investigate the possibility of using the user's address book (contacts list) to continuously authenticate the device to ensure the services are only accessed the mobile phone that belongs to the legitimate user. We call this authentication the *Who-You-Know (WYK)* scheme. For our research, we developed three components, the *WYK-Mobile-Service*, *WYK-API-Server* and a *Mobile-Demo-Application*. The *WYK-API-Server* exposes a set of authentication server APIs and the *WYK-Mobile-Service* consumes these APIs to authenticate the device every time the mobile applications are launched and make a request to the API server. Finally, the *Mobile-Demo-Application* will extract user's data from the server if the device is successfully authenticated.

Keywords: device authentication, mobile authentication, address book.

1 Introduction

Authenticating a user on a mobile phone is an essential task as users store sensitive personal information (such as photos, text messages and call history data), financial details (like banking application and their associated data) and other essential applications' data. Smart phones implement two separate authentication mechanisms. These are the native device and application authentication mechanisms. The native device authentication, is part of the device's operating system/platform. The aim is to verify the user's identity before accessing any software or hardware features of the mobile device. On

the other hand, application authentication aims to verify users before they are allowed to access the applications features.

Different devices and applications employ different authentication mechanisms depending on their security requirements. Some of the common native device authentication approaches in mobile phones are Personal Identification Number (PIN), passwords, graphical patterns and biometrics data (such as fingerprint and facial). These native authentication approaches are often used when users launch the device and they require the user's active participation. However, the main problem with PIN and password approaches is that users usually choose a simple combination that can be easily guessed by imposters to gain access into the user's device [1]. On the other hand, graphical patterns leave traces of oily residues on the screen that can easily be repeated to unlock the target device [2]. The use of biometrics is considered as the most secured among these authentication mechanisms [3]. However, according to [4], biometrics are hard to keep secret, stolen biometrics pose lifelong security risks to users as they cannot be reset and re-issued. Furthermore, transactions authenticated by biometrics across different systems are linkable and traceable back to the individual identity (in other words they don't preserve the user's privacy). These device authentication mechanisms, have three Major challenges. Firstly, device activity is permitted on an all-or-nothing basis, depending on whether the user successfully authenticates at the beginning of a session. This ignores the fact that tasks performed on a mobile device have a range of sensitivities, depending on the nature of the data and services accessed. Secondly, users are forced to re-authenticate frequently due to the bursty nature that characterizes mobile device use. Owners react to this by disabling the mechanism, or by choosing a weak "secret". Thirdly, none of these mechanisms authenticate the device from which the services are being accessed.

In addition to the devices' native authentication, mobile applications use the popular username-password combination to authenticate their users. However, the use of biometrics is also recently getting a lot of acceptance. The common challenge of mobile application authentication is the user's identity is only verified only one time during the application setup process. To compensate for this most security critical applications perform a one-time-password for consecutive interactions with the application features [5]. However, still the user is not continuously authenticated during these transactions and offer little towards transparent user authentication [6]. As a solution a number of continues authentication mechanisms are proposed over the years [7,8,9]. The common factor among these mechanisms is that they aim to track the user's usage patterns and then use it to continuously verify the identity of the user throughout the application session. However, their usability is often glossed over. Usability along with security is another important factor that plays a pivotal role in evaluating user authentication schemes. This leads to an important question, how to trade-off between security and usability? [10]. The usability of an authentication mechanisms is one of the dominant attributes that influence users' acceptance of a particular authentication scheme [11]. Even though, continuous authentication sounds secure and attractive, it has not been practically implemented due to usability factor.

In this paper, we investigate a novel mobile device authentication mechanism based on the uniqueness of an address book. An address book is ubiquitous among all mobile

devices and considered be unique to each person. In addition to that, an address book is dynamic, which is an important characteristic for authentication mechanisms. In our work we investigate the possibility of using this entity (an address book) to successfully and continuously authenticate the mobile device from which services are accessed. This mechanism can be used, in conjunction with the existing authentication mechanisms, to address the above mentioned challenges. Furthermore, it can also be used to bind services to devices.

The rest of the paper is organised as follows. Section 2 provides an explanation of the different ways and mechanisms of authentication an entity (a user or a device). This is followed by section 3, which presents a survey of published authentication mechanisms particularly for mobile device platforms. Section 4 provides a detailed discussion of the proposed WYK authentication mechanism. This includes a detailed discussion of the proposed protocols, such as the *Initialization*, *Update* and *Authenticate* protocols . Section 5, gives a full analysis and implementation of the proposed WYK authentication mechanism. Analysis details include how often address books changes and a proof-of-concept implementation of it. Finally, Section 6 concludes the paper and offers future recommendations.

2 Authentication on Mobile Devices

Entity (user or device) authentication is performed in different ways by different systems. All of them, however, have to have an identification and verification information. In this section, we discussed the nature of these information and the different, yet commonly deployed, authentication mechanisms on mobile devices.

2.1 Ways to Authenticate a User

Ways a computer system authenticates a user are broadly categorized into three categories [12], these are the "what-you-know", "what-you-have" and "what-you-are".

1. **What-you-know:** The "What-you-know" is a category of authentication mechanisms that uses a secrete knowledge (an information that a legit user only knows) to verify the identity of the claimant. The most widely used "what-you-know" authentication mechanism is the use of PINs, passwords or graphical patterns. For such mechanisms to work, the users sets up the secrete knowledge prior to using the authentication mechanism, which is commonly known as the enrollment process. Two of the most common challenges of such a mechanism are (1) the selected secrete knowledge must be easily rememberable, and (2) it must be difficult for others to easily guess it.
2. **What-you-have:** The "what-you-have" category refers to authentication mechanisms that use physical devices that the legitimate user possesses. Such mechanisms use smart cards, special hardware tokens or even mobile phones to verify the identity of the user. Usually, these devices store a unique identification/verification information that is then presented to the verifier by bringing them into contact or simply into proximity with it.

3. **What-you-are:** The mechanisms that fall into the last and third category, "what-you-are", relies on the measurement of the users physiological characteristics, such as fingerprint, facial and other biometric features. On smartphones, physiological traits, like facial features and fingerprints, can be collected using the built-in hardware, i.e. camera and finger scanners. However, other forms of biometric data , such as iris or retina require special hardware modules. Similarly, behavioral biometric data, such as gait, swipe, touch, and voice can be profiled unobtrusively, by collecting user data using various built-in mobile phone sensors (such as accelerometer, gyroscope, magnetometer, proximity sensor, touch screens, and microphone) [13].

2.2 Authentication Mechanisms

Researchers and solution providers have been investigating on how to use and/or combine the different ways of authentication (discussed in Section 2.1) in order to design and develop different authentication solutions for different application scenarios. Some of the most common authentication mechanisms are:-

1. **One-Shot Authentication:** One-shot authentication is a type of authentication mechanism in which users' credentials are verified only at the beginning of the session [14]. This is a simple process where the user claims an identity and provide his or her credentials, and the identity claimed is only accepted, by the verifying system, upon successful verification of the provided credentials. On smartphone, such credentials can be a secret knowledge (like PINs, passwords, graphical patterns, etc) or biometric data (fingerprints, facial, voice, etc). Under such mechanism the session remains valid until the user or the system closes it.
2. **Repeated Authentication:** Periodic authentication is simply the variant of "one-shot authentication" in which idle timeout duration is set, for closing the session, automatically [14,7,15]. If a user remains inactive for more than the idle timeout duration, the device locks itself. Once the session closed, new session is created by re-authenticating the user again.
3. **Continuous Authentication:** As the name implies, continuous authentication mechanisms are developed to authenticate a legitimate owner throughout the entire session. Unlike the *one-shot* or *repeated* authentication continuous authentication mechanisms, continue to verify the identity of the user after a successful point-of-entry authentication until the session ends. If any anomaly is detected by the device, the session will be stopped and all existing granted access will be terminated, immediately, and the device requests for an explicit re-authentication [7,8,9].
4. **Transparent Authentication:** Transparent authentication is a concept that stresses more on the procedure of collecting and analysing user authentication identifiers [16]. Specifically, if the system performs authentication steps in the background, without requiring explicit user cooperation [16,17]. This concept is also called implicit or unobtrusive authentication systems.
5. **Multifactor Authentication:** Multifactor authentication utilizes the concept of combining 2 or more authentication ways, that is, e-mail verification, OTP via SMS, phone call to the predefined numbers, push notification to the paired device, smart tokens, and so on, along with the usual method of authentication [18,19]. A very

common practice is registering ones mobile number with service providers, and whenever the corresponding user accesses that service for sensitive operation, for example, online banking, service provider sends the one-time passcodes (OTPs) via SMS, getting assured that a legitimate user has requested access to that service.

3 Related Work

As mentioned above, smartphone operating systems and platforms come with native authentication mechanisms, such as the use of PINs, passwords and graphic patterns. Even though these mechanisms provide the first line of defence in authenticating users, they have a number of drawbacks. In addition to the drawbacks discussed above, they are also vulnerable to shoulder surfing. To address this, Alexander De Luca et al. [20], proposed the use of the back of the mobile device to input authentication patterns. Although such approach protects the device from shoulder surfing attacks, it still inherits the same drawbacks as the other native authentication mechanisms.

To address these inherited challenges and drawbacks a number of behavioural biometrics based continuous authentication mechanisms are proposed [21,22,7]. In their paper, Özlem Durmaz Incel et al. [21] discussed the use of behavioral biometrics to complement the usual point-of-entry authentication. In their work, users' behavioural biometrics (interactions with the touch screen, such as micro-movements, pressure, finger movements, etc.) were collected using mobile phones integrated sensors (such as accelerometer, gyroscope, and magnetometer) and then used to continuously authenticate the user. Their experiment yield a 99% true positive recognition rate (TPR) and 3.5% equal error rate (EER). Their continuous authentication method was implemented on a live banking application and authentication data was collected from 45 users of the bank. In another but similar work, Pin Shen Teh et al. [22] user touch screen dynamics is used to reduce user impersonation, in case the built in 4 digit PIN was compromised, from a 100% to merely 9.9% by applying both the 4 digit PIN authentication and touch screen dynamics together. Similarly in [7] they have used touchscreen gesture data to continuously authenticate mobile phone users. Even though these mechanisms open the possibility of using behavioral biometrics as authentication information among mobile users, the challenges are; (1) these mechanisms only target to verify the identity of the users not the devices, and (2) their practical usability is often glossed over.

Another approach to authenticate devices that has been proposed and discussed in the literature is the use of mobile phone together with additional sensor devices to authenticate a target device. In [23], the authors proposed the use of body area network to provide device-to-device authentication. In their work, they utilized devices worn on the same body to collect acceleration patterns to continuously authenticate the devices. Similarly, in [24] the authors investigate a proximity based IoT device authentication that uses movement of near by host mobile device. These approaches, however, inherit the same challenges as the behavioral biometrics mechanisms discussed above regarding their usability. Essentially they collect and analyse similar information and will have similar challenges.

Recently, a different approach have gained momentum in the device authentication arena and that is the used blockchain to verify devices at real-time [25,26,27,28]. The

basic mechanics of this approach is to store immutable device information, such as serial numbers, in an open and yet secure digital ledger and, then use it to authenticate the devices on demand. Although such approach is resistant to unauthorised changes and certainly solves the usability issue of the mechanisms discussed above, it requires prior device registration. This could lead to practical issues when users change their devices. Another security weakness of this approach is its use of static authentication information. This could possibly introduce a security vulnerability into the system as it could lead to false positive results if the information used falls in the hands of malicious actors.

4 WYK Authentication Scheme

A mobile phone's address book (contacts list) is expected to be unique. This is mainly because of the different contact circles (family, work, friends, foreign correspondents, etc) in one's address book. In this paper we investigate the possibility of using this unique address book to authenticate mobile devices on demand. The proposed device authentication scheme involves the following three main entities;

1. **WYK-API-Server:** this is an API server that implements the device authentication logic. The API server stores the device's unique authentication information extracted from the address book. Depending on the privacy requirement the API server may also store other personal information and share it with the applications during and/or post authentication. The server exposes all device authentication functions as a RESTFul APIs for the other entities to consume.
2. **WYK-Service:** this is a mobile service locally installed on the target mobile device. This mobile service acts as a gateway between the API server and the target applications. Up on completion of device authentication, the *WYK-Service* broadcasts the authentication result to all the applications that registered for this service.
3. **Mobile Application:** this entity is a mobile application that requires device authentication service.

The scheme defines at three authentication phases; the (1) Initialization, (2) Update and (3) Authentication phase. Before we dive into the details of the protocols, it is worth mentioning all the protocol messages are tunnelled within https link to provide confidentiality and integrity of exchanged messages.

4.1 Phase 1: Initialization

The initialization phase, the first phase, is performed when the *WYK-Service* is installed on the target mobile phone. The primary purpose of this phase is to setup the mobile device and the *WYK-API-Server* for future authentication tasks. During installation, the service generates its pair of asymmetric keys (Pub_k^m, Prv_k^m). The Pub_k^m is the public and Prv_k^m the private key. After the keys are generated, the target device then executes the initialization protocol as illustrated in Figure 1.

The protocol is started by the *WYK-Service* sending its public key (Pub_k^m) to the *WYK-API-Server*. Then the server responds by generating a random challenges (*Rand*)

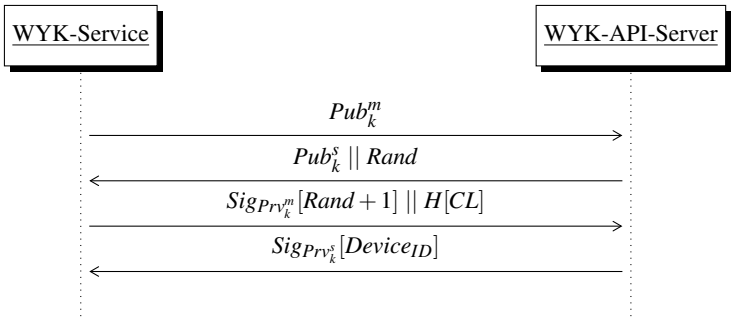

Fig. 1: Initialization protocol of the WYK Service

and sending it back along the server's public key (Pub_k^s). At this point both entities have each other's public keys. The *WYK-Service* then increments the random challenge, compute hash value of the address book, sign them and send them to the *WYK-API-Server*. Up on successful verification of the signature, the server generates a unique identity, signs it and forwards it to the service. The protocol is then concluded by the service by verifying the signature and initializing the *WYK-Service* with the received unique ID.

4.2 Phase 2: Update

One of the challenges of using the device's address book is its dynamic nature. User's address book changes often and when it does the *WYK-Service* and the *WYK-API-Server* need to keep track of the changes. The update phase is a synchronization process between the service and the API server.

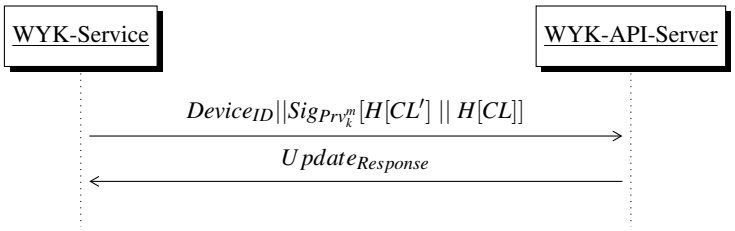

Fig. 2: WYK scheme update protocol

As illustrated in Figure 2, whenever the address book data is changed, the service computes new address book hash value, signs it and sends it to the server along with the unique device id (established during the initialization phase), and the old hash value. The server then verifies the signature and compares the old hash value received and the hash value in its database. If the old hash value comparison checks out the servers

updates it with the new hash value, otherwise rejects the update request. Finally, notifies the service about the update phase result.

4.3 Phase 3: Authentication

This phase is the device authentication phase. At this stage, the *WYK-Service* and *WYK-API-Server* have their public key pair, each other's public key, unique device identity information ($Device_{ID}$) and hash value computed over the device's address book ($H(CL)$). As illustrated in Figure 3, the authentication phase is triggered by mobile application requesting to authenticate the device.

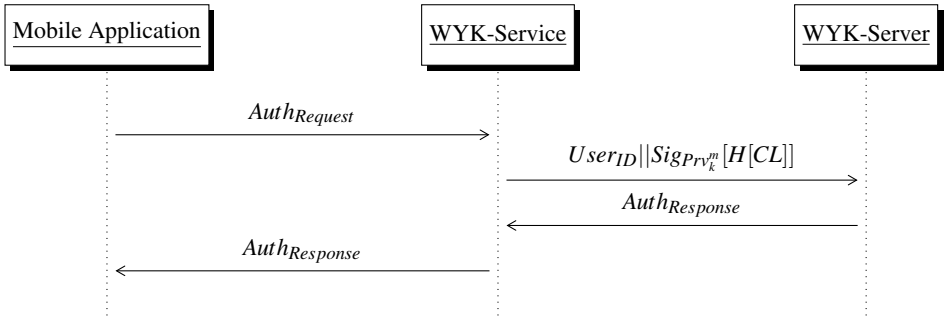

Fig. 3: WYK scheme authentication protocol

Up on reception of the request, the service forwards the device ID and signed address book hash value to the server. The server verifies the signature and the received hash value. If it is valid logs the request and responds with authentication response ($Auth_{Response}$). The *WYK-Service* then concludes the authentication process by relaying the authentication response to the requesting mobile application.

5 Implementation and Analysis

In our work we have worked on two different types of implementation of the proposed WYK scheme. Firstly, we have implemented it on Android platform which has three separate entities of the scheme. Secondly, Analysis of the proposed scheme is presented against criteria to evaluate the scheme's suitability for authentication purposes.

5.1 Proof of Concept Implementation

The proof-of-concept implementation has three separate entities and these are; *WYK-API-Server*, *WYK-Service* and a *Mobile-Application*. The *WYK-API-Server* is a back-end component of the system that has a simple database (mySQL [29]) and an application server that exposed four API end-points. The database stores authentication data such as address book hash value, unique identity number, hash update time and device

authentication time. On the other hand, the application server implements the proposed scheme's protocols and exposes them as restful APIs. The APIs are:

- */wyk/api/v1/user/initialize*: is called only once at the beginning of the scheme. It generates unique identity for the device and populates the database with the initial address book hash value.
- */wyk/api/v1/user/update*: this API is called when ever the address book is modified. Accordingly it updates the relevant database fields with the freshly computed hash value.
- */wyk/api/v1/user/authenticate*: this end-point is called when ever device authentication is required. It performs the authentication as defined in the protocol and responds with positive or negative authentication result.
- */wyk/api/v1/user/info*: finally, this end-point, returns the update and authentication request log when invoked. This API can only be invoked after successful devices authentication and only returns data belonging to a particular device. The aim of this API is to simulate how application may use the proposed scheme.

The *WYK-API-Server* is implemented using Golang [30] and screenshot of the application running is presented in Figure 4.

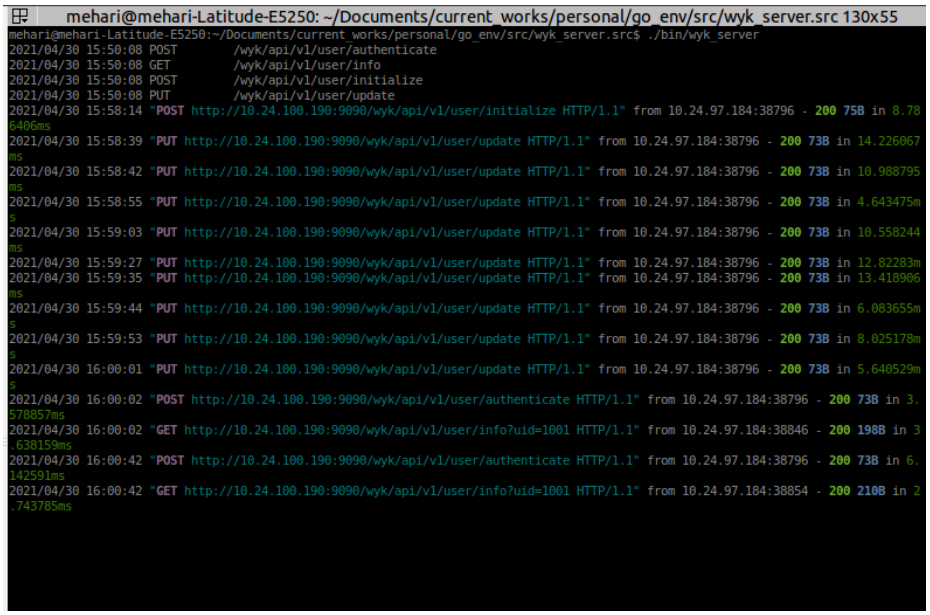

Fig. 4: WYK API Server.

The *WYK-Service* is a mobile application that consumes the APIs exposed by the *WYK-API-Server*. This entity is basically a background mobile service implementation that provides a link between the mobile device and the back-end server. It also provides

a service intent link to applications on the target device. For instance, when an application is launched, it sends device authentication intent to the service and the service invokes the authenticate end-point with the required data, and broadcasts back the result. It also implements a content observer class to keep track of changes to the address book data set. Whenever change is detected the service update the back-end server by calling the update end-point. Similarly, when the service is installed and started for the first time it initializes the back-end server as described earlier. Screenshot of the service is presented in Figure 6.

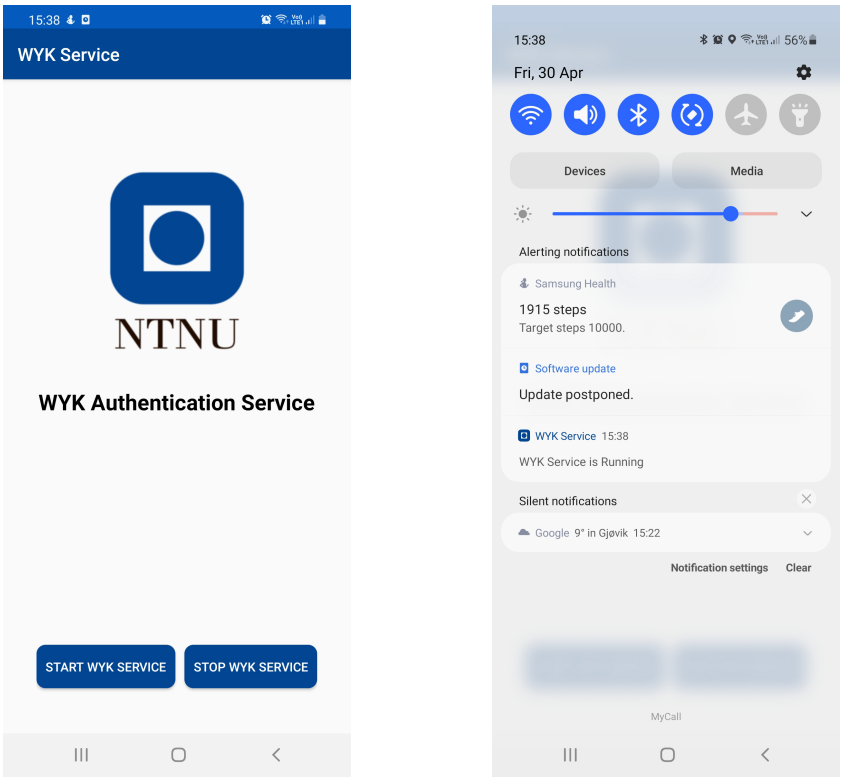

Fig. 5: WYK Service.

The last entity in our proof-of-concept implementation is the *Mobile Application*. The mobile application was designed to demonstrate how the device authentication scheme can be used. When the application is launched, it first requests device authentication and waits for the result. The result is received via a broadcast receiver. Up on a successful authentication result, the application then proceeds to the next screen and fetch log data (hash value update and authentication request time) that belongs only to the target device. Again screenshot of our mobile application is shown in Figure 6.

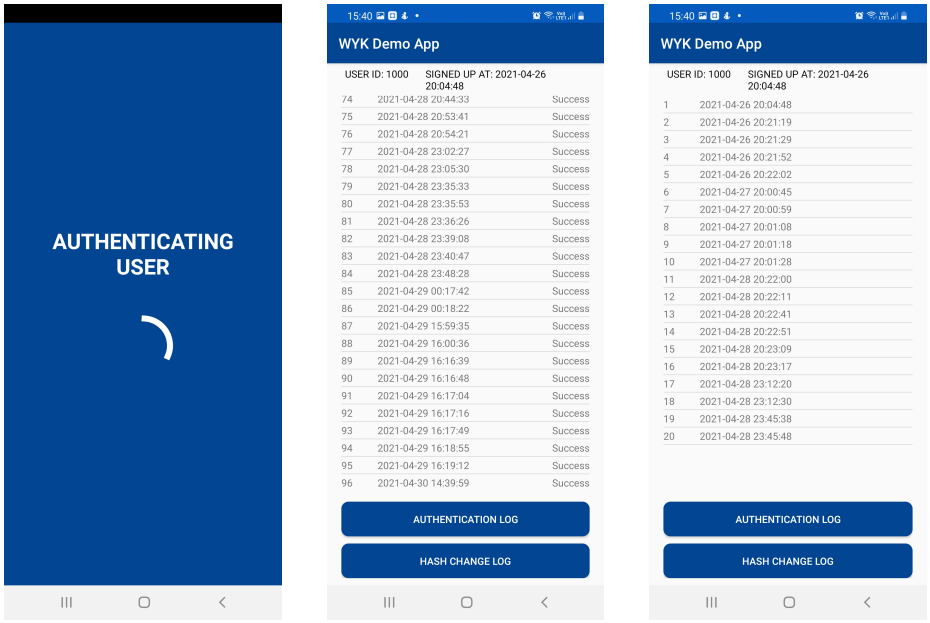

Fig. 6: A mobile application that consumes the device authentication service.

5.2 CasperFDR: Formal Verification

Another implementation is the casper-fdr implementation of the proposed protocols. Casper is a program that will take a description of a security protocol in a simple, abstract language, and produce a CSP description of the same protocol [31,32]. The CSP description is then fed into the FDR [33] for security analysis. FDR is used to either find attacks on protocols, or to show that no such attack exists, subject to the assumptions of the Dolev-Yao Model [34] (i.e. that the intruder may overhear or intercept messages, decrypt and encrypt messages with keys that he knows, and fake messages, but not perform any cryptographic attacks on the underlying protocols).

** casper implemetation of the protocols and the CSP analysis using the FDR platform will be update here. Currently working on the platform setup – will take couple of days.

5.3 Analysis

Any authentication scheme have an identification and verification information. The only requirement for an identification information is to be unique within the realm of the authentication system. Example of an identification information is email address, user name, etc. However, the verification information must satisfy a number of criteria to be considered as a viable solution. Our proposed WYK authentication scheme is evaluated against the following criteria.

1. **Universality:** Universality refers to a basic question of ”how many of the target devices have the required authentication data/information?”. The WYK scheme uses mobile phone address book as its verification information which is present in 100% of smartphones regardless of the type and model of the device.
2. **Stability:** Stability refers to whether the authentication information is static or changes over time. Some of the most popular authentication information, like PIN, passwords and cryptographic keys, are static. This however, can be a vulnerability if the authentication information is easy to guess or falls into the wrong hands. Thus, the need to change them regularly or introduce a challenge-and-response into the authentication schemes that use them to introduce dynamism. An address book is a dynamic data set that has two parts; (1) the raw contacts data (like name, number, etc) and, (2) generated data (such as last time it was changed, last time a particular contact was contacted, how many times, etc). This makes it a dynamic data set that changes multiple times a day. Sometimes, during our test, a given address book changed up to 40 times a day, which meant new authentication information was computed 40 times during that day.
3. **Collectability:** Another critical question that needs answering when evaluating an authentication mechanism is ”how easy is the information to collect?”. This hugely affects the usability of the mechanism. Sometimes an information can fulfill all the requirements but proves impractical if it can’t be collected and processed by the target devices. The WYK scheme only requires a read access to the device’s address book and processing it is quite trivial as it only means executing a cryptographic hash algorithm.
4. **Performance:** Does the information have achievable authentication accuracy, speed and memory requirements? Although such metrics can vary from device to device based on their computational capacity, WYK scheme takes less than 5 seconds to compute the hash (on Samsung Galaxy S20 with an address book with 800+ entries). Verifying the hash is typically done in milliseconds.
5. **Acceptability:** Acceptability is another critical evaluation point that determines the success of any authentication scheme. Anything that requires access to contacts list can create unease among users. However, the proposed scheme only computes a hash value over them. The fact that hash values are irreversible should comfort users that there won’t be any privacy issues for using our scheme.
6. **Forgery Resistance:** Last but not least is the question of ”how easy is it to replicate the required authentication information and fool the system?”. The WYK scheme uses the address book in its entirety to extract the authentication hash value. Thus, it would take someone to replicate the entire address book of a target. However, since it is dynamic and changes depending on the calling and/or SMS patterns of the user, an adversary must replicate the state of a given address book at particular time to successfully fool the system.

6 Conclusion

This paper has explored a novel approach of using mobile address book, which is ubiquitous among mobile phone devices, as a dynamic authentication information to provide

device authentication service. Our proposed scheme have three protocols to manage the device authentication life-cycle. In our work we have implemented the proposed authentication scheme on Android platform to provide a working principle of the scheme. Finally, we have provided analysis of the scheme against a number of authentication mechanism evaluation points.

References

1. Vishal M. Patel, Rama Chellappa, Deepak Chandra, and Brandon Barbello. Continuous user authentication on mobile devices: Recent progress and remaining challenges. *IEEE Signal Process. Mag.*, 33(4):49–61, 2016.
2. Adam J. Aviv, Katherine L. Gibson, Evan Mossop, Matt Blaze, and Jonathan M. Smith. Smudge attacks on smartphone touch screens. In Charlie Miller and Hovav Shacham, editors, *4th USENIX Workshop on Offensive Technologies, WOOT '10, Washington, D.C., USA, August 9, 2010*. USENIX Association, 2010.
3. Liam Mayron. Biometric authentication on mobile devices. *IEEE Security & Privacy*, 13(03):70–73, may 2015.
4. Mozhgan Azimpourkivi, Umut Topkara, and Bogdan Carbunar. A secure mobile authentication alternative to biometrics. In *Proceedings of the 33rd Annual Computer Security Applications Conference*, page 28–41, New York, NY, USA, 2017. Association for Computing Machinery.
5. Mohammadreza Hazhirpasand Barkadehi, Mehrbaksh Nilashi, Othman Ibrahim, Ali Zakeri Fardi, and Sarminah Samad. Authentication systems: A literature review and classification. *Telematics and Informatics*, 35(5):1491–1511, 2018.
6. Mohammed Abuhamad, Ahmed Abusnaina, DaeHun Nyang, and David A. Mohaisen. Sensor-based continuous authentication of smartphones' users using behavioral biometrics: A survey. *CoRR*, abs/2001.08578, 2020.
7. T. Feng, Z. Liu, K. Kwon, W. Shi, B. Carbunar, Y. Jiang, and N. Nguyen. Continuous mobile authentication using touchscreen gestures. In *2012 IEEE Conference on Technologies for Homeland Security (HST)*, pages 451–456, 2012.
8. Issa Traore and Ahmed Awad E. Ahmed. *Continuous Authentication Using Biometrics: Data, Models, and Metrics*. University of Victoria, Canada, 2011.
9. Mario Frank, Ralf Biedert, Eugene Ma, Ivan Martinovic, and Dawn Song. Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication. *IEEE Trans. Inf. Forensics Secur.*, 8(1):136–148, 2013.
10. Christina Braz and Jean-Marc Robert. Security and usability: the case of the user authentication methods. pages 199–203, 01 2006.
11. Mary Frances Theofanos, Ross J. Micheals, and Brian C. Stanton. Biometrics systems include users. *IEEE Systems Journal*, 3(4):461–468, 2009.
12. Fred B. Schneider. Something you know, have, or are. `https://www.cs.cornell.edu/courses/cs513/2005fa/NNLauthPeople.html`.
13. Nils Forsblom. Were you aware of all these sensors in your smartphone? `https://blog.adtile.me/2015/11/12/were-you-aware-of-all-these-sensors-in-your-smartphone/`, 2015.
14. Sandeep Gupta, Attaullah Buriro, and Bruno Crispo1. Demystifying authentication concepts in smartphones: Ways and types to secure access. *Advances in Personalized Mobile Services*, 2018.
15. Elisa Bertino, Claudio Bettini, Elena Ferrari, and Pierangela Samarati. An access control model supporting periodicity constraints and temporal reasoning. *ACM Trans. Database Syst.*, 23(3):231–285, 1998.

16. Heather Crawford and Karen Renaud. Understanding user perceptions of transparent authentication on a mobile device. *Journal of Trust Management*, (7), 2014.
17. Attaullah Buriro, Bruno Crispo, and Yury Zhauniarovich. Please hold on: Unobtrusive user authentication using smartphone's built-in sensors. In *2017 IEEE International Conference on Identity, Security and Behavior Analysis (ISBA)*, pages 1–8, 2017.
18. Kevin Michaluk, Phil Nickinson, Rene Ritchie, and Daniel Rubino. The future of authentication: Biometrics, multi-factor, and co-dependency. `https://www.androidcentral.com/talk-mobile/future-authentication-biometrics-multi-factor-and-co-dependency-talk-mobile`, 2021.
19. Mark Stanislav. *Two-Factor Authentication*. IT Governance Ltd., 2015.
20. Alexander De Luca, Emanuel von Zezschwitz, Ngo Dieu Huong Nguyen, Max-Emanuel Maurer, Elisa Rubegni, Marcello Paolo Scipioni, and Marc Langheinrich. Back-of-device authentication on smartphones. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, page 2389–2398, 2013.
21. Özlem Durmaz Incel, Seçil Günay, Yasemin Akan, Yunus Barlas, Okan Engin Basar, Gülfem Isiklar Alptekin, and Mustafa Isbilen. DAKOTA: sensor and touch screen-based continuous authentication on a mobile banking application. *IEEE Access*, 9:38943–38960, 2021.
22. Pin Shen Teh, Ning Zhang, Syh-Yuan Tan, Qi Shi, Wee-How Khoh, and Raheel Nawaz. Strengthen user authentication on mobile devices by using user's touch dynamics pattern. *J. Ambient Intell. Humaniz. Comput.*, 11(10):4019–4039, 2020.
23. Dominik Schürmann, Arne Brüsch, Stephan Sigg, and Lars Wolf. Bandana — body area network device-to-device authentication using natural gait. In *2017 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 190–196, 2017.
24. Jiansong Zhang, Zeyu Wang, Zhice Yang, and Qian Zhang. Proximity based iot device authentication. In *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, pages 1–9, 2017.
25. Meng Shen, Huisen Liu, Liehuang Zhu, Ke Xu, Hongbo Yu, Xiaojiang Du, and Mohsen Guizani. Blockchain-assisted secure device authentication for cross-domain industrial iot. *IEEE Journal on Selected Areas in Communications*, 38(5):942–954, 2020.
26. Umair Khalid, Muhammad Asim, Thar Baker, Patrick C. K. Hung, Muhammad Adnan Tariq, and Laura Rafferty. A decentralized lightweight blockchain-based authentication mechanism for iot systems. *Clust. Comput.*, 23(3):2067–2087, 2020.
27. Liangqin Gong, Daniyal M. Alghazzawi, and Li Cheng. Bcot sentry: A blockchain-based identity authentication framework for iot devices. *Information*, 12(5), 2021.
28. Fulong Chen, Yuqing Tang, Xu Cheng, Dong Xie, Taochun Wang, and Chuanxin Zhao. Blockchain-based efficient device authentication protocol for medical cyber-physical systems. *Security and Communication Networks*, 2021.
29. MySQL. Mysql: The world'd most popular open source database. `https://www.mysql.com/`.
30. Golang. Go. `https://golang.org/`.
31. Gavin Lowe. Casper: a compiler for the analysis of security protocols. In *Proceedings 10th Computer Security Foundations Workshop*, pages 18–30, 1997.
32. University of Oxford: Department of Computer Science. Installing casper. `http://www.cs.ox.ac.uk/gavin.lowe/Security/Casper/installation.html`, 2021.
33. University of Oxford. Fdr4 - the csp refinement checker. `https://cocotec.io/fdr/`.
34. Jonathan Herzog. A computational interpretation of dolev-yao adversaries. *Theor. Comput. Sci.*, 340(1):57–81, 2005.