Received September 15, 2019, accepted September 30, 2019, date of publication October 22, 2019, date of current version November 8, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2948922

Multifactor Authentication Protocol in a Mobile Environment

MACIEJ BARTŁOMIEJCZYK[1], EL FRAY IMED[1], AND MIROSŁAW KURKOWSKI[2]
[1]Faculty of Computer Science and Information Technology, West Pomeranian University of Technology, 70-310 Szczecin, Poland
[2]Institute of Computer Science, Cardinal Stefan Wyszynski University in Warsaw, 01-815 Warszawa, Poland

Corresponding author: El Fray Imed (ielfray@zut.edu.pl)

This work was supported by the West Pomeranian University of Technology.

ABSTRACT The implementation of services that process confidential data in a mobile environment requires an adequate level of security with the strictest possible mechanisms of information protection. The dominance of mobile devices as client applications of distributed systems has led to the development of new techniques that combine traditional methods of protection with protocols leveraging the potential of numerous interfaces available from a smartphone. For this reason, an upward trend in the use of biometrics-based methods and dynamically generated OTP secrets can be observed. Mobile devices are increasingly used in complex business processes that require strong user authentication methods, which, according to the European Commission (Regulation), must use at least two authentication factors belonging to different categories. Therefore, on the basis of the analysis of the solutions presented so far, a distributed protocol has been proposed. It enables user authentication using three authentication factors: possession, knowledge, and inherence. The described authentication scheme refers to the possibility of carrying out the process in the mobile environment of the Android platform with guaranteed authentication support.

INDEX TERMS Authentication, electronic identification, electronic transactions, mobile platform, multi-factor authentication, smartphone, three-factor authentication.

I. INTRODUCTION

One of the mechanisms of increasing the level of security of authorized access to resources relies on the use of multifactor authentication processes. According to the Commission Implementing Regulation (EU) 2015/1502 [1], a strong authentication process involves the use of at least two authentication factors belonging to different categories. This criterion indicates that strong authentication systems can use any combination of factors: {knowledge, inherence}, {knowledge, possession}, {possession, inherence}, or all them {knowledge, possession, inherence}. Similar requirements are presented in NIST publications [2], [3]; ''Electronic Authentication Guideline'' [2] and ''Digital Identity Guidelines'' [3] describe a high security level of authentication process based on two authenticators (authentication factors). All of these documents indicate a relationship between the security level and the number of authentication factors used.

Systems that use mobile devices as client applications that access private resources can use multi-level user identity protocols. The specificity of using mobile devices entails an increased risk of losing control of a device that is a key point of the authentication infrastructure. Due to their size and the environment in which smartphone users go through the authentication process, the classic methods of authentication are insufficient. One way to increase the security level of the authentication process may be to seek the optimal implementation of two-factor authentication. Not all security mechanism and protocols are suitable for the mobile environment. Even the implementation details and differences between Android and iOS platforms could have a major impact on the security gaps in the proposed protocol. As a result, the design of security protocols should be considered within the capabilities of the specific environment.

In order to ensure the highest possible efficiency of authentication, it is necessary to:

1) Consider both the potential of combining and implementing pairs of factors: {knowledge, possession}, {knowledge, inherence}, or {possession, inherence}, as well as the ways of implementing individual factors resistant to known attacks and threats, in the specific mobile environment that has been chosen—based on mechanisms and availabilities.

The associate editor coordinating the review of this manuscript and approving it for publication was Luca Chiaraviglio.

2) Consider using more than two authentication factors (multifactor authentication). This option increases the security level of the user authentication process by increasing the number of required factors to at least three. Here, the most crucial aspect is to use authentication factors from different groups.

The aim of the present research was to analyze the possible implementations of multifactor authentication and choose the best authentication factor from each category to use in a mobile environment. The result is an authentication protocol that uses best practices, Android Platform APIs, and security mechanisms. It has to be emphasized that authentication security is affected not only by the quantity of factors used, but the method of implementation as well.

II. RELATED WORKS

The process of authentication using a mobile device can be carried out in many ways. These methods are divided into three basic groups [4]: static password authentication, dynamic password authentication, and biometric authentication.

Static password authentication is a well-known method of user authentication. It is a process of verification whether the person who wants to be authenticated knows the correct data passed during registration, i.e. login and password. However, even simple processes like static password authentication can be implemented in many ways, depending on the implementation environment. For example, the Android Platform enables developers to use pattern, pin, or alphanumeric password [5].

Dynamically entered passwords are implemented as OTP (One-time password). The most popular implementation methods provide the secret by sending SMS messages to a trusted number associated with the user. Unfortunately, this method is characterized by a high susceptibility to phishing attacks, whose effectiveness, depending on the content of the sent message, may reach 60% of attempts [6]. This shows that many implementations are insecure; the OTP protocol should use a channel that protects the sending of secrets and enables the user to verify the identity of the sender.

The confirmation of user identity using biometrics is possible in many ways. Biometric authentication can be divided into two groups [7]: physiological biometric authentication and behavioral biometric authentication.

Examples of behavioral biometrics methods have been described in the literature [7], [8]. These schemes took into account not only the pattern of stored screen coordinates, but also the typical features of user interaction, such as the pressure force or delay between entering logging data. Due to the frequency of actions taken by users of mobile devices, numerous attempts have been made to use behavioral biometric authentication to enable on-line authentication. The main objective is to determine constantly whether the user is logged in or not. One of the ways of implementation is the use of a touch screen and slide operation events [9]. The data are extracted and classified using OCSVM (One-class Support Vector Machine) and iForest (isolation forest). Behavioral biometrics can also be used in situations where there are no standard interfaces such as keyboards. Such environments include, for example, IoT devices, which can be worn by the user. Staying in constant contact with the user enables the establishment of a pattern that can be used for authentication, for example based on movements of the hand. Appropriate extraction of features allows one to determine the user's credibility and identity [10]. Another possibility for on-line authentication involves a system analyzing the typical activities performed by the user: walking, sitting, getting up, running, or going up and down the stairs [11]. Creating models describing these activities is possible thanks to data from an accelerometer, gyroscope, and magnetometer. The authentication process periodically compares memorized patterns with the data received from the sensors. Correct classification can be achieved by using classifiers, such as SVM (Support Vector Machine), DT (Decision Tree), and K-NN (k nearest neighbors). Online authentication is also possible in combination with physiological biometric authentication [12]. For example, the characteristic features of the face obtained with a smartphone camera can be used as a source of reference data. In order to exclude well-known attacks, the implementation assumes the use of a robust liveness detection module.

Depending on the implementation, each of the presented methods is vulnerable to certain attacks. For example, static passwords are threatened by password-guessing attack, while their mobile counterparts using screen patterns can be broken by shoulder-surfing attack or smudge attack. Errors in the implementation of biometric methods result in increased susceptibility to direct attacks and indirect attacks. Some publications have shown that gaps can be effectively exploited within the framework of a sensor attack or replay attack [13]. Dynamic authentication with the use of a one-time password [6] may also be weakened at the level of the transport layer. For example, an attacker with good knowledge of the telephone network may be able to use Caller ID spoofing attacks, resulting in unreliable SMS messages reaching the user [14].

One way to increase the level of security of mobile systems is to combine individual authentication factors into multifactor systems. One of the proposed solutions relies on the connection of static logging data with biometric voice and face recognition. This protocol takes into account the connection of user data with data used for device identification (IMEI/IMSI) [15]. User credentials and biometry were used in the next authentication scheme, which assumes obtaining two-way authentication [16]. This scheme generates hashes of values obtained on the basis of reader biometric features. It is assumed that there is no need to store biometric-feature patterns in the database. On the basis of read values, the system generates hashes related to session keys and pseudorandom numbers. There is also an idea to connect a static password with a device identified by an IMEI number. The combination can be established with an efficient algorithm of block encryption, e.g. Blowfish. On the basis of the data entered by the user, the data of the device, and salt, the protocol determines a pair of keys that are used to encrypt the challenge. The time elapsed since the completing the first step of the user identity confirmation protocol [4] is also taken into account during authentication. These authentication processes can be performed by implementation of mechanisms used in standard client-server systems—the user's terminal is usually a computer and an additional authentication factor may be a properly prepared key, referring to the assumptions of asymmetric cryptography [17].

Other solutions rely on inherence and possession (two-factor authentication) for user authentication, for example based on face recognition and the one-time code sent by SMS message [7]. These systems implement the assumptions of both behavioral biometric authentication and physiological biometric authentication, because the received code is entered by sliding a finger across the screen. The process ends with the implementation of a handwriting recognition algorithm and extraction of features typical of the user's handwriting pattern. It is also possible to use passwords and keys generated during the registration process and stored on a specific device [18]. For example, a random hash, created on the server side to dynamically establish a pair of keys, can be displayed on the registering terminal screen in the form of a QR code. The code is then scanned by the mobile device that decodes the secret and executes the algorithm generating the keys. The key of the mobile device is stored in an encrypted form, where the key is created based on the password entered. This scheme was developed to eliminate the susceptibility to shoulder-surfing attack.

Another mechanism for confirming the user's identity involves using a Smart Card [19]. Since SC computing capabilities are limited, the developers of the solution proposed a simplified protocol for comparing biometric features that can be implemented using of devices of this class. The proposed solution aims to protect against threats resulting from irrevocability of biometric features processed during the authentication process.

Several of the thus far presented authentication schemes [7], [9], [10], [12] designed for the mobile environment focus on a single authentication factor. Thus, these protocols are not in line with the strong authentication objectives of EU Regulations [1] and NIST requirements [2] [3]. Meanwhile, the other user authentication schemes presented, although based on at least two factors from different categories of knowledge, possession, and inherence, also do not fully comply with the assumptions described in EU Regulations [1] and NIST requirements [2], [3]. In any scheme based on the pair of factors {knowledge, possession} [4], attention should be paid to the manner of implementation of the possession factor. It assumes confirmation of the possession of the device using sequences generated on the basis of static IMEI and IMSI of the device associated with the user. However, this protocol is not resistant to the Unauthorized-access attack and Smartphone-loss attack, because the proposed possession factor implementation is based on software access to fixed values and does not provide ways to verify that the smartphone owner is the real user. A similar scheme for implementing the possession factor was also in [15]. The use of fixed IMEI and IMSI values allows one to confirm that the authentication request comes from a device associated with the user during the registration process; however, these are values that can be read by any application installed on the mobile device. For other schemes, such as those presented in [16]–[19], according to section 2.3.1 of Regulation [1] relating to the authentication mechanism, a high level of trust is required from the scheme used to implement dynamic authentication. Within such a process, the factor that serves as evidence of the identity of the user must be changed whenever authentication is attempted. However, this approach has not been implemented in any of these schemes. Thus, despite the fact that, in the case of schemes presented in [4], [16], an attempt to generate session keys on the basis of IMEI/IMSI was made, they cannot be considered as factors meeting the required criteria.

Schemes that use three-factor authentication are becoming more important. There have been numerous research proposals adding one more authenticator to the two that are typically used. Some solutions assume authentication based on password, IMEI + SIM number, and biometrics [20]. This approach could make security stronger, but has also flaws due to the use of a static value like IMEI. Other proposed protocols are based on the addition of a biometrics factor and OTP code [21], [22]. Many of the three-factor authentication protocols also use Smart Cards [23]–[26] to ensure biometric privacy. These solutions compare biometric feature with a template within the SC (''match-on-card'' solutions). These protocols assume that the environment is equipped with a terminal enabling the usage of the card. Some of them mention the mobile environment as a target environment, but do not have any information regarding the details of mobile platform usage [20]–[22]. Further, the protocol described in Section III is designed for use not only in mobile applications, but also in web applications. Other solutions consider three-factor authentication in a completely different environment, requiring access to external scanners and hardware devices [23]–[26]. As a result, it is not really possible to use them in a mobile environment.

Here we propose that the combination of three authentication factors: knowledge, possession, and inherence, along with the implementation of dynamic authentication including public key cryptography in one authentication scheme can significantly reduce the likelihood of a false authentication session and, as a result, the unauthorized access to services. Our proposed scheme pursues the postulate of using a minimum of two credentials of different categories, dynamic authentication (point 2.3.1 of Regulation [1]), and protection against duplication and manipulation (point 2.2.1 of Regulation [1] relating to electronic identification means), and, therefore, it fully implements the demands of the Commission Implementing Regulation (EU) 2015/1502 [1].

III. THE PROPOSED MULTIFACTOR AUTHENTICATION SCHEME

Every designed security mechanism is only as good as its worst implementation. This is why the authentication protocol, especially in the case of multifactor authentication, should be related to the capabilities of the environment in which it will be implemented. The environment for which the present scheme has been designed is a system consisting of a client application implemented as Web Application that uses a trusted mobile device for the authentication process, supporting the process of identity confirmation. The mobile application uses the Android mobile platform. Due to the close link between the proposed scheme and the capabilities of the Android version, a minimum version of the platformAndroid 7.0 (API 24)—is required for the implementation of the described protocol. The described scheme includes three-factor user authentication using the components of knowledge, possession, and inherence. The notation used to describe the proposed protocol is presented in Table 1.

TABLE 1. Notation and parameters of protocol.

Symbol	Description
AS	Authentication server
WA	Web application
MA	Mobile authenticator
FCM	Firebase Cloud Messaging – push notification server
SUUID	Session identifier using UUID format [27]
H(•)	Secure one-way bcrypt hash function
E(k, •)	Asymmetric key encryption with key k
D(k, •)	Asymmetric key decryption with key k
S(k, •)	RSA digital signature process with key k
V(k, •)	RSA digital signature verification process with key k
p_{i1}	Password of user u_i
id_i	Identity of user u_i
h_{i1}	Result of the calculation of hash values from p_i in registration process
s_i	Random data used as an additional input to H(•)
OTP_1	One-time password generated by AS
OTP_2	One-time password received in MA and typed to WA
k_{pub_e}	Asymmetric public key corresponding to k_{pr} stored in AS and bound to u_i (used for encryption).
k_{pr_e}	Asymmetric private key corresponding to k_{pub} generated and stored in Android KeyStore (used for encryption).
k_{pub_s}	Asymmetric public key corresponding to k_{pr} stored in AS and bound to u_i (used for signature).
k_{pr_s}	Asymmetric private key corresponding to k_{pub} generated and stored in Android KeyStore (used for signature).
$fcmTok_i$	Firebase ID token, bound to u_i, enables sending notification with FCM
t_s	Current timestamp
t_{sg}	Object generation timestamp
t_{sv}	Object validation timestamp
EXP	Constant defining the validity of the authentication factor

A. SYSTEM ARCHITECTURE

The proposed scheme assumes communication between the client application, called "WA," and the server application, called "AS". Further steps of the authentication protocol use the exchange of messages between AS, Firebase Cloud Messaging, called FCM, and the mobile application, MA. The use of a mobile device significantly increases the number of interfaces that can be used for user authentication. HTTPs requests are processed using the latest version of TLS. An additional communication channel uses messages sent from AS to MA that are generated by the FCM intermediary service.

B. PREREQUISITES

The implementation of the proposed authentication protocol assumes the following initial conditions. The described requirements include data obtained during the process of user registration and system initiation; however, the user registration process is not described in the presented protocol.

1) SERVER APPLICATION

1) The application stores data of a registered u_i user, who is identified by an id_i.
2) The user data contain the digest of the password generated during the registration process using H(p, s) and associated with the u_i.
3) AS stores s_{i1} associated with u_i that was used in registration process.
4) AS stores $fcmTok_i$ associated with u_i. $fcmTok_i$ is identifier in FCM service.
5) AS stores $k_{pub_s}$and k_{pub_e}associated with u_i.

2) MOBILE APPLICATION

1) The application generates and stores two pairs of RSA keys with a length of 4096 bytes. One of the pairs, „iSIGN_KEYS," is used to implement the digital signature service, while the other, „ENC_KEYS," is used for encryption. The „SIGN_KEYS" key is marked as requiring user authentication if the user wants to use it. Additionally, the generated private keys are not exportable.
2) Due to a lack of continuity between the execution of the second and third authentication factors, the application generates two pairs of asymmetric cryptographic keys. Diversification of the keys increases the resistance of the protocol to potential attacks.
3) The application connects to the FCM, the user u_i registers in the service using the id_i that is known by AS.

C. KNOWLEDGE FACTOR

Fig. 1. shows the sequence diagram associated with providing a knowledge factor.

In order to confirm the identity, u_i enters the id_i and the password p_{i2} in the WA, which is used to generate h_{i2} hash by AS:

$$h_{i2} = H(p_{i2}, s_i) \tag{1}$$

AS searches id_i and creates the SUUID object and the $SUUID_k$ related to SUUID. t_{sg} is then set to: $t_{sg}(SUUID_k) = t_s$. The next step is to verify the stored hashes h_{i1} and h_{i2}.

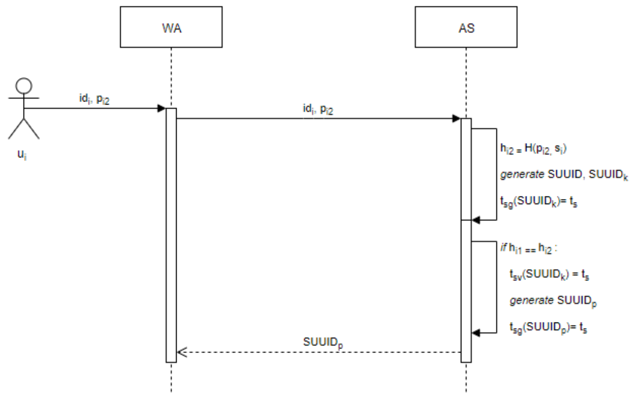

FIGURE 1. **Sequence diagram of providing knowledge factor.**

Consequently, the h_{i2} value is created in login process:

$$h_{i1} == h_{i2}. \tag{2}$$

As a positive result of the comparison, the value of t_{sv} is then assigned $t_{sv}(SUUID_k) = t_s$ and $SUUID_p$ related to SUUID is initialized. The value of t_{sg} is then assigned $t_{sg}(SUUID_p) = t_s$ and $SUUID_p$ is returned as a response to WA. It should be emphasized that the time of generating an identifier for a specific authentication factor is modified only by AS. The scheme does not provide any interfaces enabling manipulation of timestamp values, neither by MA nor WA.

D. POSSESION FACTOR

Fig. 2. shows the sequence diagram associated with providing a possession factor.

The user initiates the procedure for authentication with *possesion factor* by selecting the appropriate GUI item in the WA. $SUUID_p$ is sent to the server and used to find SUUID. The $t_{sg}(SUUID_p)$ is compared:

$$abs(t_{sg}(SUUID_p) - t_s) < EXP \tag{3}$$

If the condition for $SUUID_p$ is fulfilled, AS generates OTP_1. A value of t_{sg} is set: $t_{sg}(OTP_1) = t_s$. AS finds id_i by SUUID and $fcmTok_i$. The $fcmTok_i$ is an identifier in FCM service, related to u_i. A message containing enc_1 is delivered to MA via a secure channel within the FCM:

$$enc_1 = E(k_{pub\_e}, OTP_1) \tag{4}$$

u_i receives a message in MA and decrypts the value of enc_1 :

$$OTP_2 = D(k_{pr\_e}, enc_1) \tag{5}$$

u_i reads the value displayed on the screen of the mobile device and enters the OTP_2 value in the WA interface. When SUUID and $SUUID_p$ are found $t_{sg}(SUUID_p)$ is compared:

$$abs(t_{sg}(SUUID_p) - t_s) < EXP \tag{6}$$

If the condition is met, the values of OTP_1 and OTP_2 are compared. If the comparison is successful, the value $t_{sv}(SUUID_p)$

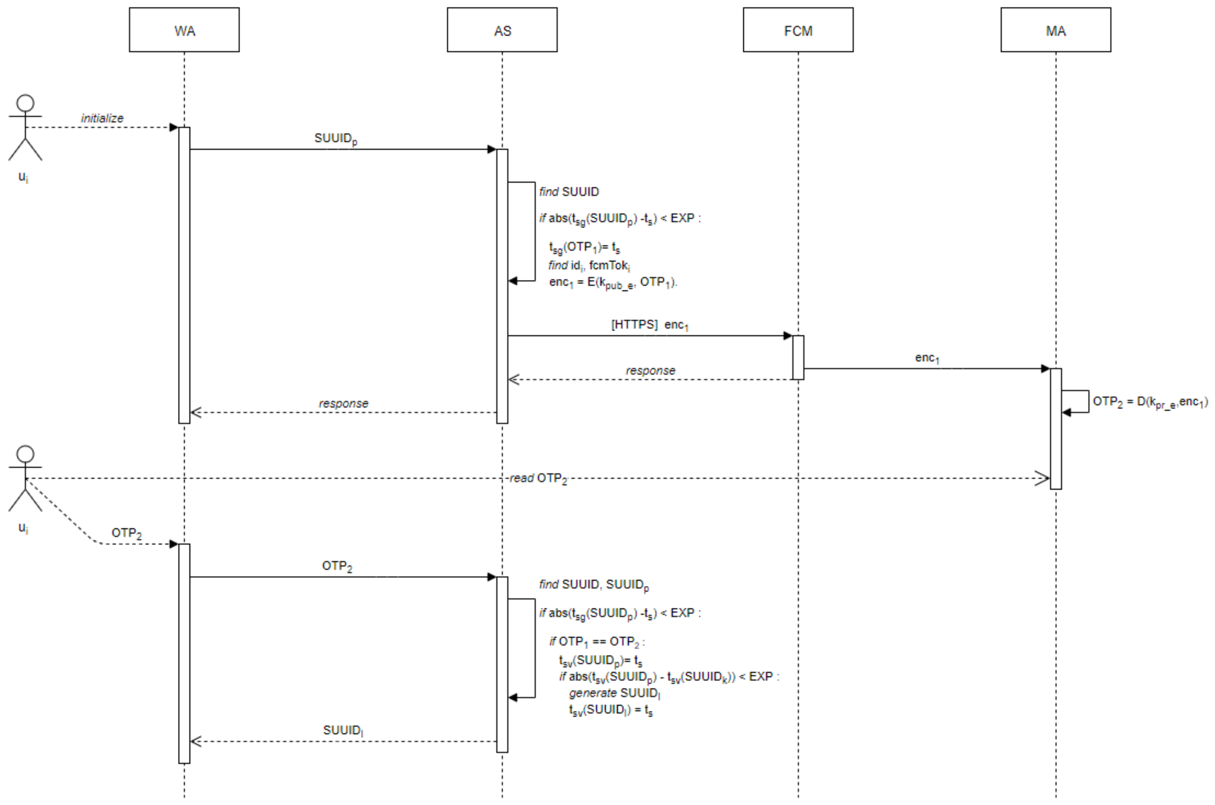

FIGURE 2. **Sequence diagram of providing possession factor.**

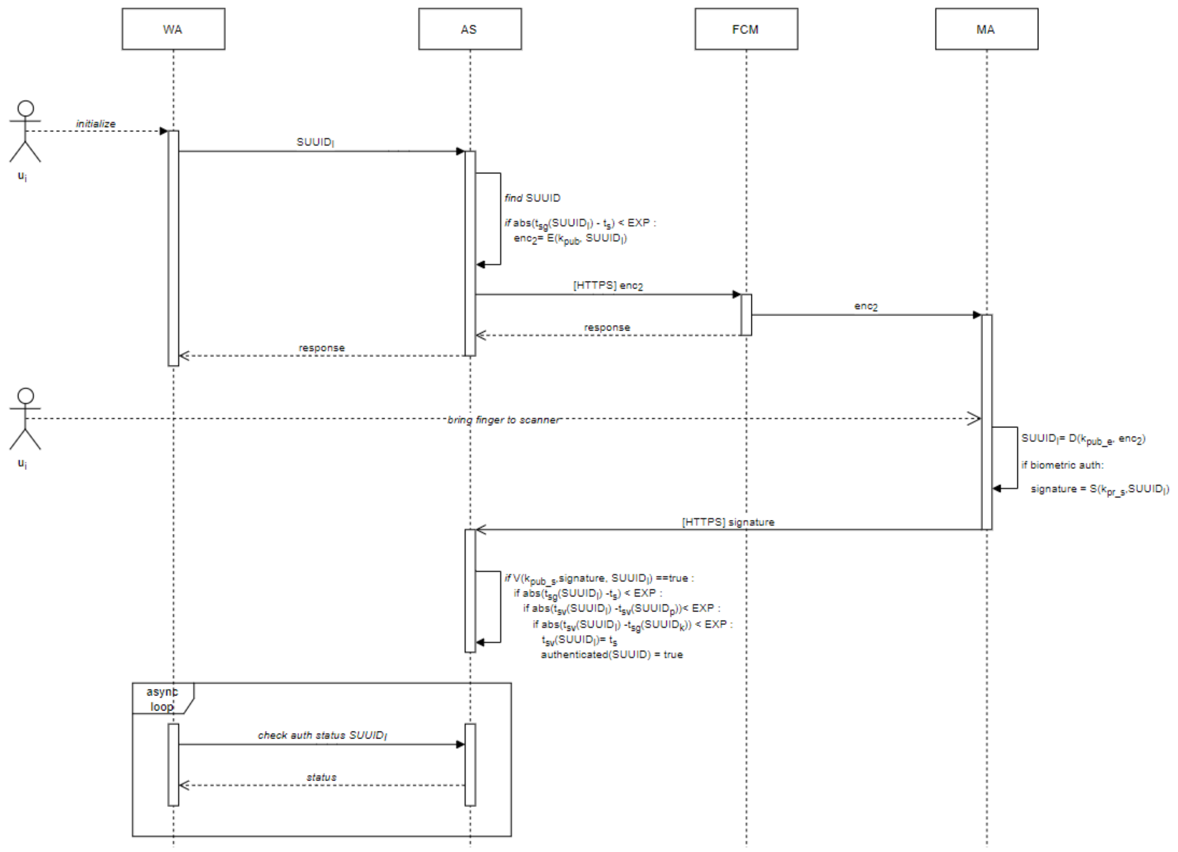
WA
AS
FCM
MA
initialize
u_i
SUUID_I
find SUUID
if abs(t_sg(SUUID_I) - t_s) < EXP :
enc2= E(k_pub, SUUID_I)
[HTTPS] enc2
enc2
response
response
bring finger to scanner
u_i
SUUID_I= D(k_pub_e, enc2)
if biometric auth:
signature = S(k_pr_s,SUUID_I)
[HTTPS] signature
if V(k_pub_s,signature, SUUID_I) ==true :
if abs(t_sg(SUUID_I) -t_s) < EXP :
if abs(t_sv(SUUID_I) -t_sv(SUUID_p))< EXP :
if abs(t_sv(SUUID_I) -t_sg(SUUID_k)) < EXP :
t_sv(SUUID_I)= t_s
authenticated(SUUID) = true
async loop
check auth status SUUID_I
status

FIGURE 3. Sequence diagram of providing inherence factor.

is set: $t_{sv}(SUUID_p) = t_s$. Finally, the $t_{sv}(SUUID_p)$ is also compared:

$$abs(t_{sv}(SUUID_p) - t_{sv}(SUUID_k)) < EXP \tag{7}$$

In case of a positive result of verification of all conditions, AS generates $SUUID_I$ related to SUUID and sets the value of $t_{sv}(SUUID_I) = t_s$. The $SUUID_I$ is then returned to WA.

E. INHERENCE FACTOR

Fig. 3. shows the sequence diagram associated with providing an inherence factor.

The authentication process based on inherence factor is initiated from the WA user interface. WA sends the $SUUID_I$ to AS. AS verifies $t_{sg}(SUUID_I)$:

$$abs(t_{sg}(SUUID_I) - t_s) < EXP \tag{8}$$

If the condition is fulfilled, a message containing an encrypted $SUUID_I$ value, enc_2, is sent to MA via FCM:

$$enc_2 = E(k_{pub\_e}, SUUID_I) \tag{9}$$

MA receives the message and executes the enc_2 decryption process to obtain $SUUID_I$:

$$SUUID_I = D(k_{pr\_e}, SUUID_I) \tag{10}$$

Next, the biometric authentication process is triggered. The user, whose pattern has been added to the system authentication API, applies a finger to the fingerprint reader. In case of positive authentication, the k_{pr_s} key can be used for asymmetric cryptography services. The received $SUUID_I$ value is signed:

$$signature = S(k_{pr\_s}, SUUID_I) \tag{11}$$

The signature value is sent to AS with $SUUID_I$. AS verifies the signature of the received message:

$$V(k_{pub}, signature, SUUID_I) == true \tag{12}$$

In case of positive verification of the signature, the time that elapsed from the moment of generating the $SUUID_I$ and the time that elapsed between the implementation of the previous authentication factor and the duration of the entire process are checked:

$$abs(t_{sg}(SUUID_I) - t_s) < EXP, \tag{13}$$
$$abs(t_{sv}(SUUID_I) - t_{sv}(SUUID_p)) < EXP, \tag{14}$$
$$abs(t_{sv}(SUUID_I) - t_{sg}(SUUID_k)) < EXP \tag{15}$$

A positive outcome of all verifications results in successful authentication using the inherence factor and sets the

$t_{sv}(SUUID_I)$ value to t_a. The flag of the authenticated SUUID object is set to true.

$$t_{sv}(SUUID_I) = t_s \tag{16}$$

According to the protocol described above, the authentication process uses the communication between WA, AS, and MA. In order to confirm the possibility of using the proposed scheme, an implementation was performed, which included a sample client application WA, AS authentication server, and mobile application MA. The key elements of the implementation, from the security point of view, include the implementation of methods responsible for generating secrets and keys, as well as their use in the processes of encryption, decryption, signing, and signature verification.

IV. PROTOCOL EVALUATION

To evaluate feasibility, performance, and advantages and disadvantages of the proposed solution, an example use case of protocol was prepared. Multifactor authentication is a crucial mechanism that should be used in every system that has to know the user identity. In the diagram below, Fig. 5, there is an example usage of the proposed protocol in an Online Banking system. It shows that the designed protocol is possible to implement and can be use in a common process, like access to bank services. Because of complexity of the proposed protocol, we further evaluate the performance of the example implementation.

A. EXAMPLE USE CASE

Diagram of the process of user authentication to bank services is illustrated in Fig. 4. The module named WA could be implemented as an Online Banking web application enabling users to manage theirs accounts. MAuth is an example implementation of MA; it is an Android Application supporting the authentication process. The application is responsible for receiving messages and implementing the second and third stages of the protocol. The authentication and verification processes, all of the encryption mechanisms, and the usage of cryptographic keys are all carried out as described in Section III. Fig 4 presents a diagram of the entire authentication process, in real implementation environment. For detailed descriptions: see above. The main aim of this attempt to create example implementation is to verify assumptions in a real environment and evaluate the advantages and disadvantages of proposed protocol. The authentication process consists of a series of steps (actions): input password, read OTP from MAuth, and use a biometric scanner in built-in the mobile. A detailed description of the example implementation is included in Appendix.

B. PERFORMANCE

Multifactor authentication is a complex process that could be demanding to implement. One of the criteria that can be helpful to evaluate protocol usefulness is the performance of the solution. However, it is not simple to measure distributed processes that need interaction with the user. There are many methods of testing, but in this case manual testing was selected.

The example implementation was enriched with a logging module that saves session information, such as the SUUID, $t_s(SUUID_P)$, $t_{sv}(SUUID_P)$, $t_s(SUUID_I)$, $t_{sv}(SUUID_I)$. On the basis of the collected data, the duration of the processes associated with inherence and possession factors was calculated. The duration of entire authentication process was also measured. The performance study was based on four-attempts of authentication to same simulated online banking system. The average duration of the authentication process was 6792,2 milliseconds. It shows that, despite the complexity of protocol and the use of public key cryptography, the authentication process is fast. It should be noted that the results may be affected by external factors, such as phone performance, network speed, and, above all, the user response rate. The influence of the user's reaction on the measured protocol performance can be seen from the time needed for the realization of the possession factor. The time spent on reading and entering the code into the Online Banking system constitutes approx. 50% of the time of the whole process. Another factor affecting performance may be the fingerprint reader technology.

TABLE 2. Results of performance research.

Statistical metrics	Values
Sample size	200
Authentication, average time	6792,2 ms
Possession factor, average time	3472,78 ms
Inherence factor, average time	1608,32 ms

Further, the proposed protocol consists of a number of steps carried out by different actors. Individual processes, e.g. signature or encryption, can be carried out using the selection of any algorithms in accordance with the assumptions. For this reason, it is not possible to unambiguously determine the time and memory complexity of the authentication protocol. We decided to analyze the mobile application (MAuth) that requires the largest amount of interaction with the user. The collected data (Fig. 5) shows that the application does not exceed 130MB of operating memory of the device and thus should not be a problem for modern smartphones.

Another important security issue, accessibility, is hard to achieve in a distributed process like the proposed multifactor authentication scheme. Ensuring the highest possible availability is necessary on several layers. Considering the Online Banking (WA) and Authentication server, it is possible to use well-known scaling mechanisms [28], [29]. In stateless applications, each step of process can be performed in a different node. This ensures that, in the case of an attack on system availability, the next step can be carried out. Ensuring the security of the mobile application (MA) is a much more difficult issue. It refers to the flawless implementation of the application and, above all, to the awareness of the user

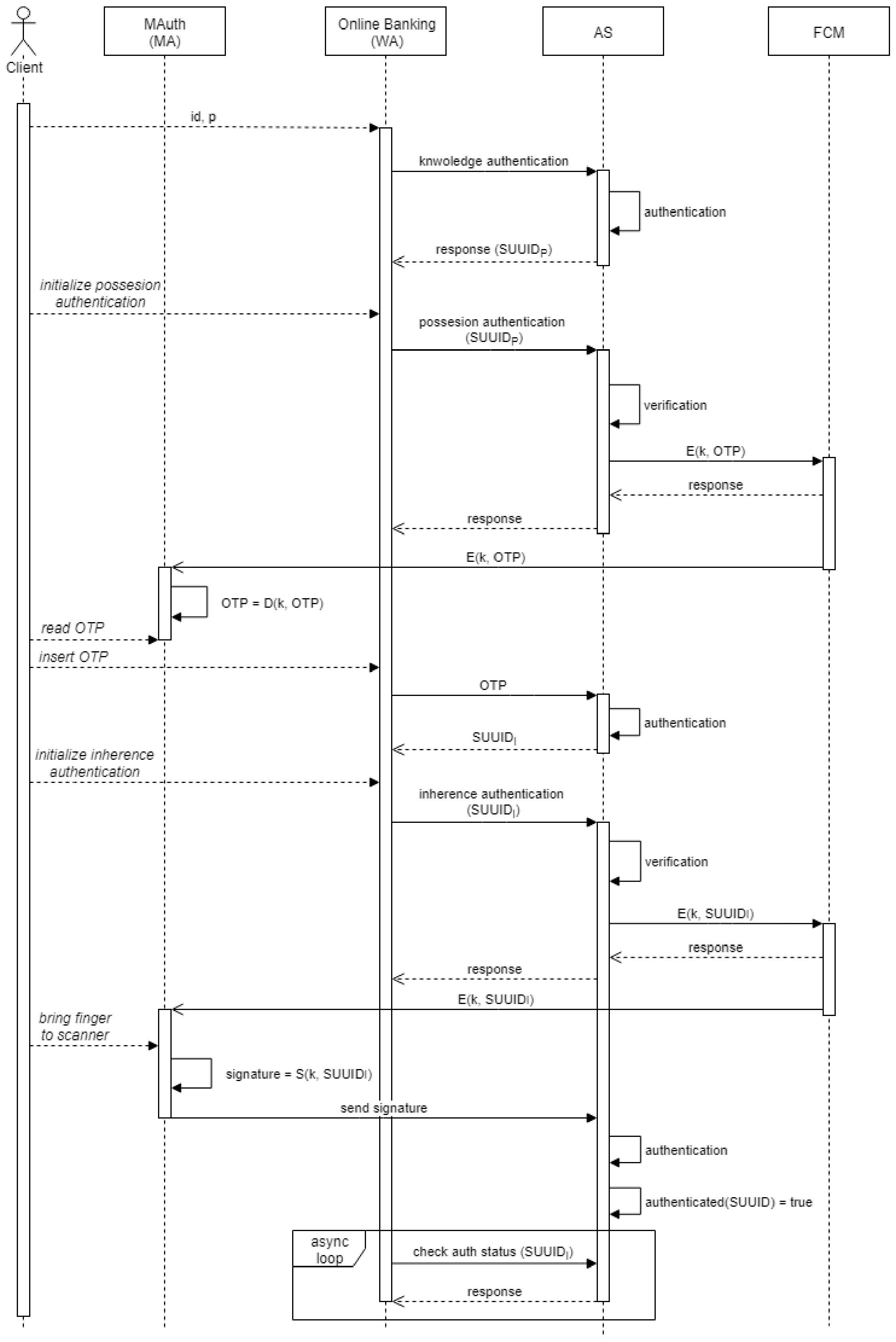

FIGURE 4. Sequence diagram of example use case.

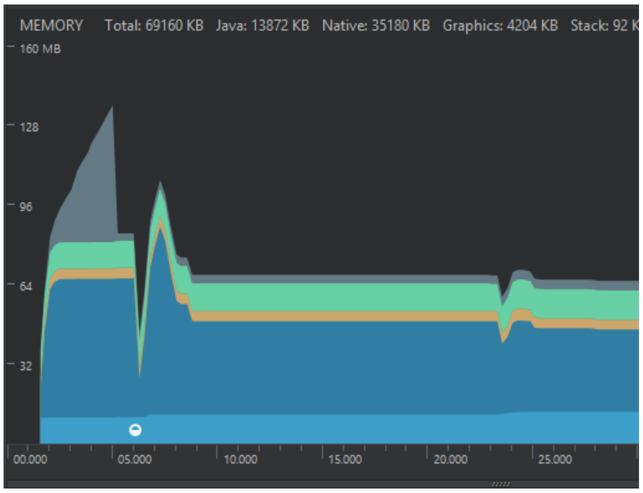

FIGURE 5. The use of operational memory by mobile application (MAuth).

using the smartphone. In order to ensure the highest possible level of protection, it is recommended to implement the application in accordance with accepted standards and recommendations [30], [31].

V. PROTOCOL VERIFICATION

Developing a suitable security protocol is not an easy process. The developer needs to consider not only the goals that the protocol should guarantee in the system, but also the properties of the protocol's working environment, i.e. its technical environment. In our work, we are dealing with a communication protocol that operates on the Internet between a trusted server representing a bank and a user/client. Also, it is important to note, that the system needs to use a properly-secured Public Key Infrastructure, whose security is assumed and not considered as an additional verification purpose.

In this research, the correctness of the proposed protocol was analyzed using formal verification. We tested the protocol's security upon two different types of attacks: Man-In-The-Middle and replay attacks. Formal methods of software system verification are now the most reliable. In many cases, researchers can find bugs or flaws in such systems and protect them against them. Sometimes it can be formally proven that the system is safe from the considered point of view. It is important to note that Man-In-The-Middle attacks include not only attacks on the confidentiality of sensitive data sent over the network during the protocol execution, but also attacks involving the impersonation of the honest user by an intruder.

For verification of our protocol it was essential to specify the assumptions imposed on the tested system. First of all, it was assumed that the perfect cryptography condition is met, i.e. the Intruder cannot decrypt the ciphertexts without having the appropriate key. It was also assumed that system components used are programmed accordingly and provide secrecy for all confidential primitives. The initial assumptions of the system's operation should also be determined. We assume that the user knows his/her ID and his/her password and that he/she has a mobile device equipped with a cryptographic module containing private keys and a biometric reader. It was also assumed that the server has all the information needed to conduct further communication (i.e. password, salt, keys, etc.).

Nowadays, model checking is the most modern and most effective method of formal verification of security protocol properties [32]–[38]. Generally, this method consists of constructing a formal model of the execution of the examined protocol in the form of a transition system (i.e. graph, automaton, Kripke structure, etc.). In these systems, the tested property is expressed often as the reachability of certain states (e.g. states corresponding to undesirable situations in the system). For these solutions, fast methods of state reachability testing were developed, which gave effective methods of testing the protocol properties.

Of course, like any other research method, model checking requires some assumptions regarding the system being built. In the case of protocols, a model of protocol execution is built in a precisely defined, finite space of cryptographic primitives and system users. Most often, the behavior of several users, including the so-called Intruder who wants to deceive others or break into the report, are considered, modeled, and investigated. These users are assigned appropriate cipher keys and other necessary primitives. An important element of these considerations is the set of permissions/assumptions granted to the Intruder. Here, the so-called Dolev-Yao model is most commonly used, which assumes that the Intruder has free access to the network but cannot decrypt ciphertexts without having the appropriate cryptographic keys [39].

Based on these assumptions, systems for automatic provision of the protocol properties are constructed and tested. To date, four recognized and well-documented systems have been developed and implemented: AVISPA [33], Scyther [34], ProVerif [38], and VerICS [35]. These tools have been used to examine many protocols, both academic examples and real systems. They are also still used to study currently developed solutions [40].

Timestamps have been used for many years to increase the a protocol's security. The protocols that use them are called time protocols. Although adding time to the considerations is very important from the point of view of the studied properties, building mathematical models that allow expressing the passage of time is not easy. Different approaches can be used in this respect, but the most appropriate discretization of continuous time is used most often [41].

In our work, we have used a method designed for automatic verification of systems modeled by Timed Automata [35]. This method has been described in detail [36] and was successfully applied in several verifications of real systems [40], [42]. This system enables investigating time properties of concurrent systems, modeled as a network of synchronized timed automata. In security protocol verification, we usually consider a finite space of users, timestamps, and other cryptographic primitives. Here, we can combinatorically generate possible executions of the investigated protocol; each execution is modeled as a so-called execution automaton. These automata are synchronized by labels with automata that model user knowledge. All of these automata create a product automaton (networks of labels synchronized by automata). Runs in this network are then encoded as boolean propositional formulae. Because of the large size of the formulae obtained, we need to use SAT-solvers that in many cases can compute whether the formula is satisfiable/the searched state can be reached. Knowing the kind of attacks considered, we can check whether an undesirable state in the system can be reached or not. If yes, this means that an attack exists. The protocol being examined here is a protocol consisting of three components related to each other by a global timestamp. It guarantees the consistency of calculation time and that there is no other task. Protocol execution starts with sending a password to the server by the user using a web application; then, the message is exchanged in the next steps of the protocol.

The whole protocol can be written in the Common Language as follows:

$$
\begin{aligned}
&\alpha_1 : A \rightarrow S : \langle i(A), N_A \rangle_{K_{AS}}\\
&\alpha_2 : S \rightarrow A : \langle t_s \rangle_{K_{AS}}\\
&\alpha_3 : A \rightarrow S : \langle t_s \rangle_{K_{AS}}\\
&\alpha_4 : S \rightarrow A : \langle N_S, t_{N_S} \rangle_{K_A}\\
&\alpha_5 : A \rightarrow S : \langle N_S \rangle_{K_{AS}}\\
&\alpha_6 : S \rightarrow A : \langle t_s' \rangle_{K_{AS}}\\
&\alpha_7 : A \rightarrow S : \langle t_s' \rangle_{K_{AS}}\\
&\alpha_8 : S \rightarrow A : \langle \langle N_S{}' \rangle_{K_A''}, t_{N_S{}'} \rangle_{K_A{}'}\\
&\alpha_9 : A \rightarrow S : \langle N_S{}', t_{N_S{}'} \rangle_{K_{AS}}
\end{aligned}
$$

Because this specification does not contain some parameters essential from the verification point of view, it is necessary to add generated primitives and the time conditions imposed on the next steps of the protocol for time specification information. The syntax of the ProToc language allows us to express this information [43].

In the area of security protocol specifications, there are several different approaches to protocol description appropriate for the purpose of automatic verification. These approaches, often simply called specification languages, are almost all related to the automatic verification tools in which they are used. The degree of complexity of these specifications varies, starting from very complicated and not intuitive [44], through quite intuitive, but sometimes redundant or complicated as HLPSL [33], to simple and intuitive, as SPDL for Scyther [34] or our ProTOC [43].

As we have mentioned already, the vast majority of these solutions were designed for time-independent protocols. The theoretical quality of these languages should also be mentioned here. Only for some, have appropriate formal semantics been built to confirm their correctness and expressiveness; the literature provides a few good examples [33], [34], [45]. The ProTOC language used in our considerations also has developed denotative formal semantics; the relevant theory has already been published [36].

Knowing the effectiveness of our method, we decided to use it for verification and specification of the proposed protocol.

The protocol specification in this language is as follows:

$$
\begin{aligned}
&\alpha_1 : \left[A; S; \langle i(A), N_A \rangle_{K_{AS}}\right], \quad [t_1; \{K_{AS}, N_A\}; \{N_A\}; \mathit{true}]\\
&\alpha_2 : \left[S; A; \langle t_s \rangle_{K_{AS}}\right], \quad [t_2; \{K_{AS}, t_s\}; \{t_s\}; t_s - t_2 \leq L_F]\\
&\alpha_3 : \left[A; S; \langle t_s \rangle_{K_{AS}}\right], \quad [t_3; \{K_{AS}, t_s\}; \emptyset; t_s - t_3 \leq L_F]\\
&\alpha_4 : \left[S; A; \langle N_S, t_{N_S} \rangle_{K_A}\right],\\
&\qquad \left[t_4; \left\{N_S, K_A, t_{N_S}\right\}; \left\{N_S, t_{N_S}\right\}; t_s - t_4 \leq L_F\right]\\
&\alpha_5 : \left[A; S; \langle N_S \rangle_{K_{AS}}\right], \quad [t_5; \{K_{AS}, N_S\}; \emptyset; t_s - t_5 \leq L_F]\\
&\alpha_6 : \left[S; A; \langle t_s{}' \rangle_{K_{AS}}\right],\\
&\qquad \left[t_6; \left\{K_{AS}, t_s{}'\right\}; \left\{t_s{}'\right\}; t_s - t_6 \leq L_F \wedge\right]\\
&\alpha_7 : \left[A; S; \langle t_s{}' \rangle_{K_{AS}}\right],\\
&\qquad \left[t_7; \left\{K_{AS}, t_s{}'\right\}; \emptyset; t_s - t_7 \leq L_F \wedge t_s{}' - t_7 \leq L_F\right]\\
&\alpha_8 : \left[S; A; \langle \langle N_S{}' \rangle_{K_A''}, t_{N_S{}'} \rangle_{K_A{}'}\right],\\
&\qquad \left[\begin{array}{l} t_8; \left\{N_S{}', K_A', K_A'', t_{N_S{}'}\right\};\\ \left\{N_S{}', t_{N_S{}'}\right\}; t_s - t_8 \leq L_F \end{array}\right]\\
&\alpha_9 : \left[A; S; \langle N_S{}', t_{N_S{}'} \rangle_{K_{AS}}\right], \quad \left[t_9; \left\{K_{AS}, N_S{}', t_{N_S{}'}\right\}; \emptyset;\right.\\
&\qquad \left. t_s - t_9 \leq L_F \wedge t_s{}' - t_9 \leq L_F \wedge t_s{}'' - t_9 \leq L_F\right]
\end{aligned}
$$

For example, let us describe the second step of the protocol:

$$\alpha_2 : \left[S; A; \langle t_s \rangle_{K_{AS}}\right], \quad [t_2; \{K_{AS}, t_s\}; \{t_s\}; t_s - t_2 \leq L_F]$$

As we can see, the specification consists of two tuples. The first tuple contains information about the sender, recipient, and the structure of the message being sent. This part of the specification does not differ from the Common Language specification. The second tuple contains information about the execution time t_2, elements needed by the sender to compose the message $\{K_{AS}, t_s\}$, elements generated for this step and further execution of the protocol $\{t_s\}$, and the time condition that must be met to perform this step $t_s - t_2 \leq L_F$. These types of conditions guarantee the resistance of the protocol to repetitive attacks.

We performed our experimental research on a computer equipped with I7-8850U, 1.80 GHz x 8 processor, 8 GB of RAM, and Ubuntu Linux operating system, with the kernel version 4.15.0-43. Using the methodology mentioned above [36], we obtained 18 different executions that model different user behaviors, including the Intruder. In our formal model, we have 18 timed automata that model executions and 31 automata that model user knowledge. This network can be used for verification of safety upon both Man-In-The-Middle and replay attack. The difference lies in the fact that we mark other states in our network for searching. These automata create new timed automation that is a network of synchronized (timed) automata. The obtained automaton was translated into a propositional boolean formula with over 215 thousand variables. A CNF version of the formula has over 600 thousands clauses. As mentioned above, the security properties were encoded as reachability property in a set of runs of our product automaton. All of our computations showed that there was no attack upon investigated protocol in both mentioned and investigated types.

In particular, for the case of a Man-in-the-Middle attack, aimed at extracting confidential information by an Intruder, it should be noted that all important cryptographic primitives, $t_s, N_S, t_{N_S}, t'_s, N'_S, t_{N_{S'}}$, that should be kept as confidential are sent appropriately encrypted by symmetric key shared by the user A and the Server S : K_{AS} (in steps: $\alpha_1 - \alpha_3, \alpha_5 - \alpha_7, \alpha_9$) or public keys of the user A: K_A, K'_A, K''_A (in steps: α_4, α_8). Considering the assumptions described at the beginning including the perfect cryptography assumption, it is not possible to compromise this data.

In the case of a replay attack, it should be emphasized that almost all protocol steps are marked by the time moments using appropriate timestamps, $t_s, t_{N_S}, t'_s, t_{N_{S'}}$ (steps: $\alpha_2 - \alpha_4$, $\alpha_6 - \alpha_9$); . The lifetime value should be chosen depending on the reality of the network in which the protocol will work.

The properties described above have been confirmed by formal verification. In the formal model built, appropriate attacks could not exist due to the assumptions presented and discussed. Also, in the case of an attack consisting of impersonating an honest user or server under assumptions mentioned and discussed, an attack is not possible. This guarantees the use of shared keys with the Server or public keys in communication, the validity of which is confirmed by the appropriate PKI used.

To sum up, with the assumptions imposed on the system and discussed above and with proper implementation of the protocol, it is safe from all known types of attacks according to assumption presented at the beginning.

VI. DISCUSSION

The presented solution relates to the requirements of the European Commission [1]. Authentication rules are global knowledge, which is why we can find very similar requirements in NIST publications [2] [3]. These cited papers define three categories of authenticators. In one NIST publication [3], we can find a recommendation to use more than one authentication factor for the design systems with high security level. However, there is no information regarding the combination of authenticators that guarantee the highest possible security level. We also cannot find this directly in the other NIST publication [2] in the section describing authentication process, but there are also rules for the creation of authentication factors: Single-factor Token and Multi-factor Token. Overall, the guidelines [2], [3] make the demand to use multi-factor tokens—to implement factors from different groups (knowledge, possesion, inherence). Our proposed protocol uses three authenticators including each of these groups, thus we claim that it meets the required criteria. According to NIST [2], it can be said that it provides security requirements of "Level 3."

The NIST "Electronic Authentication Guideline" [2] does not differentiate security level between two and three factors token. The publication considers two-factor authentication as sufficient to achieve the highest level recognized in the document. It also presents the opinion there could be environments and applications that may require such a differentiation. Based on this document [2], we can indicate all of factors that are prone to attacks. The proposed protocol considers multifactor authentication as the process in which the claimant demonstrates the possession and control of the authenticators using a mobile device. The smartphone is a key point of the security level of the protocol. Considering the public network environment, access control to the device, and the possibility to perform common attacks, it can be said it is the case when differentiation of security levels between two- and three-factor authentication should be defined. It could be easy to find a scenario like "Smartphone loss attack," when two-factor authentication is not sufficient. This is the reason why we decided to propose the three-factor authentication protocol described in this article.

Other researchers have also been considering other multifactor authentication solutions [20]–[26]. However, most of those relating to the mobile environment are very general or incomplete. The remaining ones assume a completely different implementation environment. For example, proposed architectures using smartcard terminals and extended biometric scanners are much more difficult to implement and they cannot be used for authentication in other environments, such as web applications and other popular services. The protocol proposed in this article assumes the mobile application in authenticator module that can be used in many services like mailing application or online banking.

The main advantage of solutions presented in the related works [23]–[26] is that they guarantee biometric privacy, which may be an advantage over the solution presented in this article. The solution proposed in our protocol assumes that biometrics are implemented based on the Android API. This does not require the biometric data to be shared with other third-party applications using that protocol for an authentication.

The proposed multifactor authentication protocol is complex. While this may increase the risk and vulnerability to attacks, we decided to use proven and popular mechanisms of solution implementation. Discussing FCM security is out of the scope the article; however, it is important to mention that the service has successfully completed the ISO 27001 [46] and SOC 1 [47], SOC 2 [48], SOC 3 [49] evaluation processes. As a result, it is used in popular solutions, for example in the banking sector by ING Bank (Internationale Nederlanden Groep) [50].

In addition to the formal analysis for Man-In-The Middle and reply attacks, tests were also carried out for other popular attacks listed in Table 3. The proposed protocol was also resistant to unauthorized access attacks, password guessing attacks, and smartphone loss attacks. This is directly due to the use of three authentication factors. Taking over access to the fact that the device does not provide access to the authenticators and even if an attacker can guess the credentials he/she

TABLE 3. Comparison with other existing schemes.

Criteria	[16]	[4]	[17]	Ours
Replay attack	Yes	Yes	Yes	Yes
Man-in-the-middle	Yes	No	No	Yes
Unauthorized access attack	Yes	Yes	Yes	Yes
Password guessing attack	Yes	Yes	Yes	Yes
Smartphone loss attack	Yes	Yes	Yes	Yes
DDOS	No	No	No	Yes
Formal security proof	No	No	Yes	Yes
Meets the criteria of regulation	No	No	No	Yes

only has one of the factors. Meanwhile, resistance to DDOS attack results from the use of a scalable solution infrastructure. Additionally, it is assumed that appropriate security policies will be implemented to guarantee the resistance to DDOS attacks.

VII. CONCLUSION

The presented authentication scheme uses three-factor authentications based on knowledge, possession, and inherence factors. Using all of them makes it resistant to common attacks (see Table II) including: Man-In-The-Middle, replay attack, unauthorized access attack, password guessing attack, and smartphone loss attack. User authentication is distributed, making the protocol much stronger. In this paper, we analyzed how to implement the requirements presented in EU Regulations [1]. For this purpose, we proposed a novel scheme for user authentication, based on three authentication factors: knowledge, possession, and inherence. Related works do not entirely comply with requirements such as: strong user authentication, dynamic authentication, and resistance to activities, such as guessing, eavesdropping, replay or manipulation of communication by an attacker with high attack potential able to subvert the authentication mechanisms. All of these criteria are met in by the multifactor authentication protocol presented in this paper, which includes communication between WA, AS, and MA.

One of the strongest points of our protocol is the formal analysis presented in Section 4. It proves that the proposed three-factor authentication protocol not only meets the criteria of EU Regulations [1], but also is resistant to Man-in-the-middle and replay attacks. The special notation used to describe the protocol allowed us to carry out many tests proving the reliability of the protocol—this formal proof makes the effectiveness of the algorithm undeniable.

APPENDIX

This paper presents an example process of the use of the proposed protocol. The multifactor authentication process is performed by WA (Online Banking) and MA (MAuth); some actions are performed on the WA side and some on the MA side. An example application implementing the proposed protocol for authentication is presented below.

A. KNOWLEDGE FACTOR

User inputs the tuple, login and password, into to proper fields in Online Banking (see Figure 6).

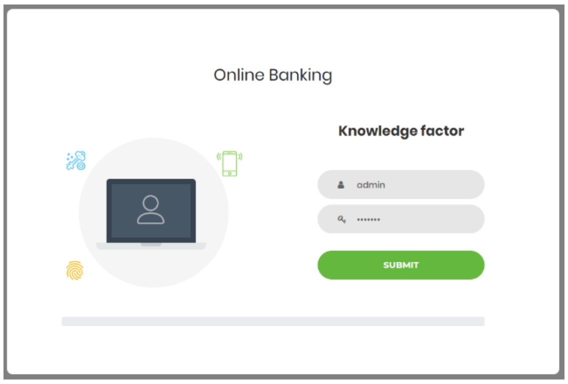
Online Banking
Knowledge factor
admin
•••••••

FIGURE 6. **Knowledge factor.**

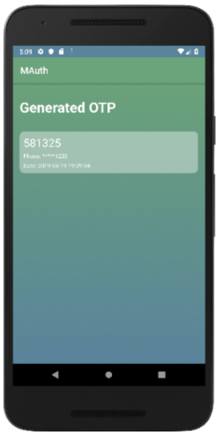
5:09
MAuth
Generated OTP
581325
Phone: *****1203
Date: 2019-08-19 19:09:04

FIGURE 7. **Possession factor – MAuth (MA).**

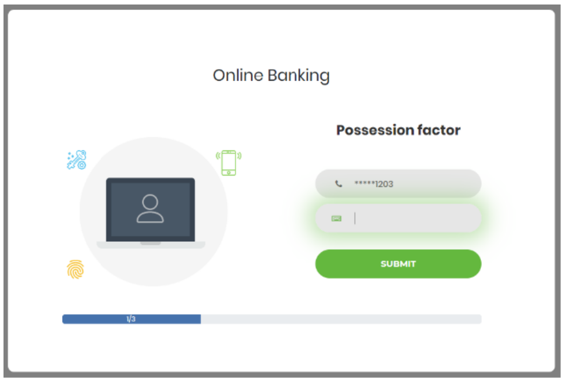
Online Banking
Possession factor
*****1203

FIGURE 8. **Possession factor – Online Banking (WA).**

B. POSSESION FACTOR

fter confirming the correctness of the password in Online Banking, an intent with the OTP code is called on the MAuth side. The received code should be entered in the proper field that appears in Online Banking (see Figures 7 and 8).

C. INHERENCE FACTOR

The last stage of authentication is to initiate the Biometric process on the Online Banking side. The appropriate intent

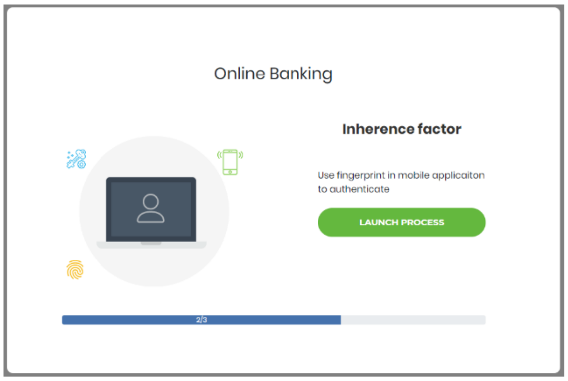

FIGURE 9. Inherence factor – (1) Online Banking (WA).

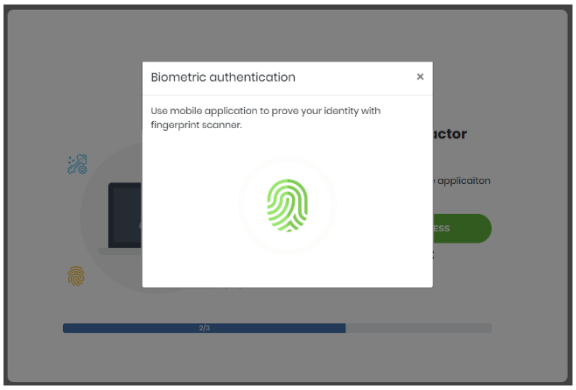
Biometric authentication
Use mobile application to prove your identity with fingerprint scanner.
2/3

FIGURE 10. Inherence factor – (2) Online Banking (WA).

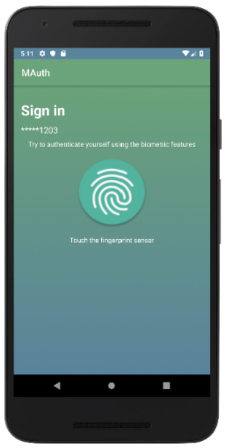
MAuth
Sign in
******1203
Try to authenticate yourself using the biometric features
Touch the fingerprint sensor

FIGURE 11. Inherence factor – MAuth (MA).

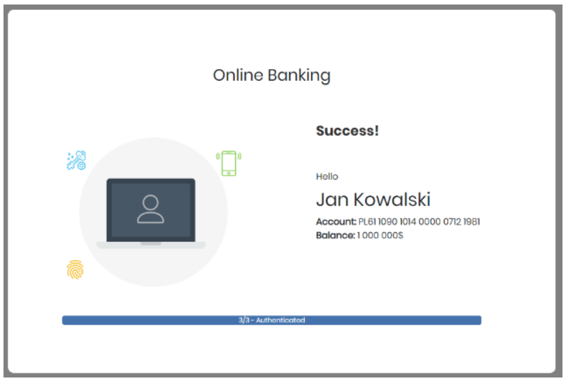

FIGURE 12. Multifactor authentication result.

is activated on the MAuth side, enabling the fingerprint to be scanned (see Figures 9-11).

After confirming the user's identity, the Online Banking screen displays sample information about the account details. Multifactor authentication has been successful (see Figure 12).

ACKNOWLEDGMENT

The authors acknowledge Mr. Włodzimierz Chocianowicz, Vice-President of PKN/TC 172 for Identification of Persons, Electronic Signature, Electronic Cards and Related Systems and Activities (Polish Standardizing Committee "mirror" technical committee of CEN TC 224 "Personal identification and related personal devices with secure element, systems, operations and privacy in a multi sectorial environment, former member of ISO/IEC JTC1 SC17 WG5 and CEN TC224 WG15 (European Citizen Card) for his consultations and valuable help.

REFERENCES

[1] *Commission Implementing Regulation (EU) 2015/1502 of 8*, European Commission, Brussels, Belgium, Sep. 2015.
[2] W. E. Burr, D. F. Dodson, E. M. Newton, R. A. Perlner, W. T. Polk, S. Gupta, and E. A. Nabbus, "Electronic authentication guideline," Dept. Commerce, Nat. Inst. Standards Technol., Gaithersburg, MD, USA, Tech. Rep. NIST SP 800-63-2, 2013.
[3] P. A. Grassi, M. E. Garcia, and J. L. Fenton, "Digital identity guidelines," Dept. Comerce, Nat. Inst. Standards Technol., Gaithersburg, MD, USA, Tech. Rep. NIST SP 800-63, 2017.
[4] Y. Yu, J. He, N. Zhu, F. Cai, and M. S. Pathan, "A new method for identity authentication using mobile terminals," *Procedia Comput. Sci.*, vol. 131, pp. 771–778, Apr. 2018.
[5] Android Help. *Support.google.com/android*. Accessed: May 2019. [Online]. Available: https://support.google.com/android/answer/9079129?hl=en
[6] H. Siadati, T. Nguyen, P. Gupta, M. Jakobsson, and N. Memon, "Mind your SMSes: Mitigating social engineering in second factor authentication," *Comput. Secur.*, vol. 65, pp. 14–28, Mar. 2017.
[7] O. Alpar, "Biometric touchstroke authentication by fuzzy proximity of touch locations," *Future Gener. Comput. Syst.*, vol. 86, pp. 71–80, Sep. 2018.
[8] T. Nguyen and N. Memon, "Tap-based user authentication for smartwatches," *Comput. Secur.*, vol. 78, pp. 174–186, Sep. 2018.
[9] Y. Yang, B. Guo, Z. Wang, M. Li, Z. Yu, and X. Zhou, "*BehaveSense*: Continuous authentication for security-sensitive mobile apps using behavioral biometrics," *Ad Hoc Netw.*, vol. 84, pp. 9–18, Mar. 2018.
[10] M. Shahzad and M. P. Singh, "Continuous authentication and authorization for the Internet of Things," *IEEE Internet Comput.*, vol. 21, no. 2, pp. 86–90, Mar./Apr. 2017.

[11] M. Ehatisham-Ul-Haq, M. A. Azam, U. Naeem, J. Loo, and Y. Amin, "Continuous authentication of smartphone users based on activity pattern recognition using passive mobile sensing," *J. Netw. Comput. Appl.*, vol. 109, pp. 24–35, May 2018.

[12] M. Smith-Creasey, F. A. Albalooshi, and M. Rajarajan, "Continuous face authentication scheme for mobile devices with tracking and liveness detection," *Microprocessors Microsyst.*, vol. 63, pp. 147–157, Nov. 2018.

[13] S. Ghouzali, M. Lafkih, W. Abdul, M. Mikram, M. El Haziti, and D. Aboutajdine, "Trace attack against biometric mobile applications," *Mobile Inf. Syst.*, vol. 2016, Mar. 2016, Art. no. 2065948.

[14] H. Mustafa, W. Xu, A.-R. Sadeghi, and S. Schulz, "End-to-end detection of caller ID spoofing attacks," *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 3, pp. 423–436, May/Jun. 2016.

[15] M. Alizadeh, S. Abolfazli, M. Zamani, S. Baharun, and K. Sakurai, "Authentication in mobile cloud computing: A survey," *J. Netw. Comput. Appl.*, vol. 61, pp. 59–80, Feb. 2016.

[16] J. Sun, Q. Zhong, L. Kou, W. Wang, Q. Da, and Y. Lin, "A lightweight multi-factor mobile user authentication scheme," in *Proc. IEEE Conf. Comput. Commun. Workshops*, Apr. 2018, pp. 831–836.

[17] J.-L. Tsai and N.-W. Lo, "A privacy-aware authentication scheme for distributed mobile cloud computing services," *IEEE Syst. J.*, vol. 9, no. 3, pp. 805–815, Sep. 2015.

[18] W. I. Khedr, "Improved keylogging and shoulder-surfing resistant visual two-factor authentication protocol," *J. Inf. Secur. Appl.*, vol. 39, pp. 41–57, Apr. 2018.

[19] R. D. Findling, M. Hölzl, and R. Mayrhofer, "Mobile match-on-card authentication using offline-simplified models with gait and face biometrics," *IEEE Trans. Mobile Comput.*, vol. 17, no. 11, pp. 2578–2590, Nov. 2018.

[20] S. Shilpa and V. Panchami, "BISC authentication algorithm: An efficient new authentication algorithm using three factor authentication for mobile banking," in *Proc. Online Int. Conf. Green Eng. Technol. (IC-GET)*, Nov. 2016, pp. 1–5.

[21] W. Kennedy and A. Olmsted, "Three factor authentication," in *Proc. 12th Int. Conf. Internet Technol. Secured Trans. (ICITST)*, Dec. 2017, pp. 212–213.

[22] B. A. Scaria and R. K. Megalingam, "Enhanced E-commerce application security using three-factor authentication," in *Proc. 2nd Int. Conf. Intell. Comput. Control Syst. (ICICCS)*, Jun. 2018, pp. 1588–1591.

[23] C.-I. Fan and Y.-H. Lin, "Provably secure remote truly three-factor authentication scheme with privacy protection on biometrics," *IEEE Trans. Inf. Forensics Security*, vol. 4, no. 4, pp. 933–945, Dec. 2009.

[24] S. M. Sujatha and Y. U. Devi, "Design and implementation of IoT testbed with three factor authentication," in *Proc. Int. Conf. Commun. Electron. Syst. (ICCES)*, Oct. 2016, pp. 1–5.

[25] H. M. Mathew, S. B. E. Raj, P. S. J. Gundapu, and S. J. F. Angeline, "An improved three-factor authentication scheme using smart card with biometric privacy protection," in *Proc. 3rd Int. Conf. Electron. Comput. Technol.*, Apr. 2011, pp. 220–223.

[26] J. Yu, G. Wang, Y. Mu, and W. Gao, "An efficient generic framework for three-factor authentication with provably secure instantiation," *IEEE Trans. Inf. Forensics Security*, vol. 9, no. 12, pp. 2302–2313, Dec. 2014.

[27] P. Leach and M. Mealling. (Jul. 2005). *A Universally Unique IDentifier (UUID) URN Namespace*. [Online]. Available: https://tools.ietf.org/html/rfc4122

[28] M. Toy, "High availability layers and failure recovery timers for virtualized systems and services," *Procedia Comput. Sci.*, vol. 114, pp. 126–131, Sep. 2017.

[29] L. Fan, W. Tiejun, and W. Liuyi, "Research and implementation on model for high availability of enterprise information system," *IERI Procedia*, vol. 3, pp. 181–185, Sep. 2012.

[30] Google. *Developer.android.com*. Accessed: May 2019. [Online]. Available: https://developer.android.com/topic/security/best-practices

[31] S. Schleier and J. Willemsen, "Mobile application security verification," OWASP, Brazil, China, Tech. Rep. NIST SP 800-63-2, 2019.

[32] W. Marrero, E. Clarke, and S. Jha, "Model checking for security protocols," School Comput. Sci., Carnegie Mellon Univ., Pittsburgh, PA, USA, Tech. Rep. CMU-CS-97-139, 1997.

[33] A. Armando, D. Basin, Y. Boichut, Y. Chevalier, L. Compagna, J. Cuellar, P. Hankes Drielsma, P. C. Heám, O. Kouchnarenko, J. Mantovani, S. Mödersheim, D. von Oheimb, M. Rusinowitch, J. Santiago, M. Turuani, L. Viganò, and L. Vigneron "The AVISPA tool for the automated validation of Internet security protocols and applications," in *Proc. Int. Conf. Comput. Aided Verification*, Jul. 2005, pp. 281–285.

[34] C. J. F. Cremers, "The Scyther tool: Verification, falsification, and analysis of security protocols," in *Proc. CAV*, Jul. 2008, pp. 414–418.

[35] M. Knapik, A. Niewiadomski, W. Penczek, A. Półrola, M. Szreter, and A. Zbrzezny, "Parametric model checking with VerICS," in *Transactions on Petri Nets and Other Models of Concurrency*, vol. 4. Berlin, Germany: Springer, pp. 98–120, 2010.

[36] M. Kurkowski and W. Penczek, "Applying timed automata to model checking of security protocols," in *Handbook Finite State Based Models and Applications*. Boca Raton, FL, USA: CRC Press, 2013, pp. 223–254.

[37] D. Basin, C. Cremers, and C. Meadows, "Model checking security protocols," in *Handbook of Model Checking*. Cham, Switzerland: Springer, 2018.

[38] A. Hess and S. Mödersheim, "A typing result for stateful protocols," in *Proc. 31st IEEE Comput. Secur. Found. Symp.* New York, NY, USA: Oxford, Jul. 2018, pp. 374–388.

[39] D. Dolev and A. Yao, "On the security of public key protocols," *IEEE Trans. Inf. Theory*, vol. IT-29, no. 2, pp. 198–208, Mar. 1983.

[40] O. Siedlecka-Lamch, I. El Fray, M. Kurkowski, and J. Pejaś, "Verification of mutual authentication protocol for MobInfoSec system," in *Computer Information Systems and Industrial Management* (Lecture Notes in Computer Science), vol. 9339. Cham, Switzerland: Springer, 2015, pp. 461–474.

[41] R. Corin, S. Etalle, P. H. Harte, and A. Mader, "Modelling and checking timed authentication of security protocols," *Fundam. Informaticae*, vol. 79, nos. 3–4, pp. 363–378, 2007.

[42] T. Hyla, I. El Fray, M. Kurkowski, W. Maćków, and J. Pejaś, "Practical authentication protocols for protecting and sharing sensitive information on mobile devices," in *Cryptography Security Systems*, vol. 448. Berlin, Germany: Springer-Verlag, 2014, pp. 153–165.

[43] M. Kurkowski, A. Grosser, S. Szymoniak, and J. Piątkowski, "ProToc—An universal language for security protocols specifications," in *Soft Computing in Computer and Information Science* (Advances in Intelligent Systems and Computing), vol. 342. Cham, Switzerland: Springer, pp. 237–248, 2015.

[44] A. Lomuscio and W. Penczek, "LDYIS: A framework for model checking security protocols," *Fundamenta Informaticae*, vol. 85, nos. 1–4, pp. 359–375, 2008.

[45] S. Briais and U. Nestmann, "A formal semantics for protocol narrations," in *Tryworthy Global Computing*. Berlin, Germany: Springer-Verlag, 2005.

[46] *Google LLC*. Accessed: May 2019. [Online]. Available: https://Firebase.google.com and https://firebase.google.com/downloads/gdpr/2019_Firebase_ISO_27001.pdf

[47] *SOC 1—SOC for Service Organizations: ICFR*. Accessed: May 2019. [Online]. Available: https://www.aicpa.org/ and https://www.aicpa.org/interestareas/frc/assuranceadvisoryservices/aicpasoc1report.html

[48] *SOC 2—SOC for Service Organizations: Trust Services Criteria*. Accessed: May 2019. [Online]. Available: https://www.aicpa.org/ and https://www.aicpa.org/interestareas/frc/assuranceadvisoryservices/aicpasoc2report.html

[49] *System and Organization Controls (SOC) 3 Report on the Google Firebase System Relevant to Security, Availability, and Confidentiality for the Period 1 November 2017 to 31 October 2018*. Accessed: May 2019. [Online]. Available: https://firebase.google.com/ and https://firebase.google.com/downloads/gdpr/2018_Firebase_SOC3_Report.pdf

[50] ING News. *My ING*. Accessed: May 2019. [Online]. Available: http://www.corporatenews.lu/ and http://www.corporatenews.lu/en/archives-shortcut/archives/article/2017/02/ing-news-push-notifications-available-on-my-ing

MACIEJ BARTŁOMIEJCZYK received the M.Sc. degree in software engineering from the West Pomeranian University of Technology, Szczecin – ZUT, Poland, where he is currently pursuing the Ph.D. degree. He participates in the design and implementation of transactional banking systems. He deals in the design of authentication protocols, especially in a mobile environment.

EL FRAY IMED received the M.Sc. degree in naval automation from the Szczecin University of Technology, the Ph.D. degree from the Faculty of Maritime Technology, and the D.Sc. degree from the Faculty of Computer Science. He is currently a Professor with the West Pomeranian University of Technology, Szczecin – ZUT, Poland. He is the author or coauthor of four book chapters and over 80 research articles on risk assessment and risk management, information systems audit, security information and event management, and common criteria.

MIROSŁAW KURKOWSKI received the M.Sc. degree in mathematics from the Jan Dlugosz University of Czestochowa, the Ph.D. degree in mathematics and computer science from the Institute of Computer Science, Polish Academy of Sciences, and the D.Sc. degree in computer science from the Faculty of Mechanical Engineering and Computer Science, Technical University of Czestochowa. He is currently a Professor with Cardinal St. Wyszynski University in Warsaw, Poland. He is the author or coauthor of four books and over 60 research articles on applied mathematical logic, formal methods in verification of software systems, especially security protocols, and cryptography.