J Supercomput
DOI 10.1007/s11227-017-2190-8

An efficient anonymous authentication protocol in multiple server communication networks (EAAM)

An Braeken[1] · Pardeep Kumar[2] · Madhusanka Liyanage[3] · Ta Thi Kim Hue[4]

Abstract In a multi-server authentication environment, a user only needs to register once at a central registration place before accessing the different services on the different registered servers. Both, from a user point of view as for the management and maintenance of the infrastructure, these types of environments become more and more popular. Smartcard- or smartphone-based approaches lead to more secure systems because they offer two- or three-factor authentication, based on the strict combination of the user's password, the user's biometrics and the possession of the device. In this paper, we propose an efficient anonymous authentication protocol in multiple server communication networks, called the EAAM protocol, which is able to establish user anonymity, mutual authentication, and resistance against known security attacks. The novelty of the proposed scheme is that it does not require a secure channel during the registration between the user and the registration center and is resistant to a curious but honest registration system. These features are established in a highly efficient way with the minimum amount of communication flows between user and server during the establishment of the secret shared key and by using light-weight cryptographic techniques such as Chebyshev chaotic map techniques and symmetric key cryptography. The performance and security of the protocol are analyzed and compared with the latest new proposals in this field.

✉ An Braeken
abraeken@gmail.com

1 INDI and ETRO Department, Vrije Universiteit Brussel, Brussels, Belgium

2 Department of Computer Science, University of Oxford, Oxford, England

3 Centre for Wireless Communications, University of Oulu, Oulu, Finland

4 School of Electronics and Telecommunications, Hanoi University of Science and Technology, Hanoi, Vietnam

Published online: 14 November 2017

Keywords Multi-server authentication · Anonymity · Chaotic maps

1 Introduction

Enterprises make more and more use of multi-server authentication environments, enabling their employees to get access with one single registration to different services running on different servers in the infrastructure. Also several e-government services can be currently accessed by the user through one single registration.

This is a significant improvement, compared with the classical authentication method where the user has to send to each of these services an identity and hashed password, which are then compared with the stored verification table at server side. Such method encompasses several security problems and requires huge storage capacities. In 2000, Hwang et al. [1] introduced the smart card-based authentication. Here, a registration center (RC) activates the card with some secret information, which is (partly) shared with the registered servers. It can be seen as a two-factor authentication since both the possession of the smartcard and the knowledge of the password is required. During the last decade, a lot of research has been performed on developing authentication protocols for this type of setting, adding more and more additional security features and becoming more and more practical. For instance, we can identify three main classes based on the used cryptographic mechanisms: (i) symmetric key based, (ii) public key based by means of elliptic curve cryptography (ECC) or chaotic maps, and (iii) bilinear pairing-based systems. However, the main drawback in a symmetric key system is that it does not allow the construction of key material without the secure channel and it offers less resistance against insider attacks. The bilinear pairings systems are highly demanding operations (in terms of time complexity, e.g., point multiplications, etc.), compared with the operations of the other classes, as shown in Table 3 of [2]. For the public key-based systems, the usage of chaotic maps is about three times more efficient than the classical elliptic curve-based operations [3]. Most of the literature provides mainly the classical security properties but requires a secure channel at the user registration, which is not practical always. Moreover, many of the schemes consist of more than three, often five different communication flows and thus enormously increase the complexity. Another vulnerability in such schemes is that a registration center could perform the required operations, honestly, but it may reveal information for its own purposes. Therefore, these schemes cannot withstand against the honest-but-curious attack. To end this, we assumed that a security protocol must satisfy the following security features in the multi-server environment:

- *Anonymity to the outsider* Identity is one of the sensitive personal information that could be leaked over the communication to track the user or device activities. Therefore, anonymity is highly required to provide the anonymous communication that can protect from such identity leaks.
- *Anonymity or identity derivation of user by the server* This security property depends on the type of service is being offered. In some cases, it is beneficial that the identity of the user is not leaked to the server for the privacy protection of

the user. The server only knows that the user is approved by the RC. However, the server should be ensured of the claimed identity of the user for the services such as payment activities.
- *Unlinkability* The multi-server communication network, two or more items of interest from an attacker's perspective, means that the attacker can sufficiently link whether these items are related to a particular user or device over the communication. Therefore, it is paramount that different messages cannot be linked to the same user by any outsider collecting the transmitted messages.
- *Mutual authentication* Both the user and the server could authenticate each other and contribute to the construction of the secret key.
- *Multiple servers* For the sake of user-friendliness, a user can access multiple servers with the same credentials, obtained during the registration phase.
- *Known key security* In the multi-server communications, it is possible that an attacker or malevolent user could determine the session key for the ongoing communication. It is therefore desirable that even if the secret session keys between the user and the server are revealed, no further information can be obtained from the private keys of user and the server.
- *Perfect forward secrecy* Even if the long-term private keys of the user or the server are leaked, the previously derived secret session keys between the user and the server cannot be revealed.
- *Resistance against known attacks* The scheme can withstand insider attack, stolen device attack, offline guessing attack, replay attack, user impersonation attack, and man-in-the-middle attack.
- *Resistance against curious but honest RC* The RC is not able to derive the secret session key generated by the user and the server and thus cannot store the sensitive information of the user. Curious means that the RC will perform the required operations, but that it may reveal or misuse information for its own benefit.
- *Avoidance of the secure channel* No secure channel is required between the user and the registration center during the registration phase.

Contribution Considering the above security properties, we propose a new and efficient anonymous authentication protocol in the multiple server communication networks (EAAM). The proposed EAAM is based on the Chebyshev chaotic map operations and required only two communication phases for the construction of the secret session key, which is quite efficient. In order to authenticate the user, the EAAM uses a smartphone with installed biometric feature identification, instead of a smartcard with a corresponding smart reader to perform the authentication. Moreover, the performance and security of the EAAM are analyzed and compared with the recently proposed literature in this field.

The rest of the paper is structured as follows. Section 2 presents related work. In Sect. 3, we give some background on the operations and the system setting. Section 4 describes our protocol. In Sects. 5 and 6, we analyse the security and performance of the protocol, respectively. Finally, we conclude the paper in Sect. 7.

2 Related work

Smartcard-based authentication mechanisms are proposed for both single server and multiple server environments [4]. They are either two-factor or three-factor based. However, as the biometrics-related information is used in the same way as the password, the two-factor-based schemes can be in most of the cases easily extended to three-factor-based schemes. In [5], a generic framework to upgrade two-factor authentication schemes to three-factor ones using the concept of fuzzy extractor [6] is explained. There exists an enormous amount of papers on these type of protocols, e.g. a detailed survey can be found in [4]. We will limit the discussion on the latest generation of multi-server authentication protocols, which offer in addition user anonymity.

We can basically distinguish three types of classes in these authentication schemes: symmetric key-based schemes, public key-based schemes, and bilinear pairing-based schemes, as follows.

Symmetric key-based schemes Many symmetric key-based schemes are also proposed in the literature [7–16]. Almost all of them have been shown to be vulnerable to one or more of the following attacks: insider attacks, impersonation attacks, and stolen smartcard attacks. Except in [8,15,16], no attacks have been mentioned yet. In [15], the proposed protocol contains three communication rounds, and in [8] the protocol is missing the unlinkability feature. The system in [16] is currently the most efficient and complete system, given the possibilities that can be obtained with symmetric key operations. For instance, these systems cannot be resistant to an insider attack where one compromised user and server collaborate. They also require a secure channel during registration, and the RC can always follow every communication between the user and the server, due to the inherent properties of symmetric key cryptography.

Public key-based schemes This is the second class where two main mechanisms are distinguished as the ECC and the Chebychev polynomials. For the ECC-based systems, we distinguish in the literature [17–25]. All these systems have been shown to be insecure [17,18,20–22,24] or perform the authentication request through the RC and thus require five communication phases [23,25]. In [19], a secure and two-round scheme is presented, but although it is mentioned as a multi-server authentication protocol, unique credentials are required for each server, making it highly inefficient. For the systems using the Chebychev polynomials as trapdoor mechanism, several schemes have been proposed that without the usage of a RC [26–35]. However, many of them are vulnerable to different types of attacks (insider attacks, man-in-the-middle attacks, etc.) as shown in [28–33]. The protocols without reported attacks in this list are currently [26,27,34,35]. For the classical setting with RC and servers, consisting of only two communication rounds, we identify the references [36–38]. The protocol in [36] is clearly not resistant against insider attacks as the RC and all the servers share the same secret key. The scheme in [37] allows to trace the user. Finally, as the RC derives in both [37,38] the secret key material of the user and the servers, both protocols are not prone against a curious RC and do require a secure channel during the registration.

Bilinear pairing-based schemes The references [26,27,39–42] belong to the third class. In [27], it has been shown that the proposed scheme was the most complete with respect to security features and computational efficiency, compared to the other bilinear protocols. Still, these types of schemes do not offer more security features as the others; all possess at least three communication phases and are very expensive due to the inherent computational demanding bilinear operation.

To conclude, as far as the authors are aware, there are currently no protocols available in literature, that satisfy besides the classical security features as anonymity, unlinkability, perfect forward secrecy, known key secrecy, that are in addition resistant to a curious and honest RC. As privacy of the user is becoming more and more important, it is an interesting feature to provide to the user. In addition, we also include the mechanisms that avoid the need for a secure channel, which is in practice difficult to realize. Moreover, as efficiency both in terms of latency, computational complexity and energy consumption is very relevant to mobile devices, it is important to limit the number of required communication rounds to a minimum and to use the most efficient security operations.

3 Preliminaries

3.1 Chebyshev chaotic map

The basics of the Chebyshev chaotic maps are now shortly discussed. For more details, we refer to [43,44]. The Chebyshev polynomials are defined as:

$$T_n(x) : [-1, 1] \rightarrow [-1, 1] : T_n(x) = \cos(n \arccos(x)), n \in \mathbb{N}$$

To enhance the properties of Chebyshev chaotic maps, Zhang [45] proposed an improved Chebyshev polynomial defined on the domain $\mathbb{R}$, where the modulo of the result is taken over a large prime number p. The following recurrent relation can be used to determine these Chebyshev polynomial maps $T_n : R \rightarrow R$ with $n \geq 2$

$$T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x) \mod p, \text{ given that } T_0(x) = 1, T_1(x) = x.$$

The improved Chebyshev polynomials satisfy the semigroup property, i.e., the equation $T_v(T_u(x)) = T_u(T_v(x)) \mod p$ holds for $u, v \in \mathbb{N}$ and $x \in \mathbb{R}$. The interesting aspect at the Chebyshev polynomials is that they can be used in the discrete logarithm (DL) problem and the computational Diffie–Hellman (CDH) problem.

DL problem Given two elements x and y, it is computationally infeasible to find an integer r such that the equation $T_r(x) = y \ (mod p)$ holds.
CDH problem Given x, $T_r(x)$, $T_s(x)$, it is computationally infeasible to compute the result of the computations $t_{rs} = T_{rs}(x) = T_r(T_s(x)) = T_s(T_r(x)) = T_{sr}(x) = t_{sr} \ (mod p)$.

Also a signature scheme [46] based on the Chebyshev polynomials has been proposed.

3.2 Usage of biometrics data

When the biometric data of a user are captured, there are inherently different types of noise included. Consequently, repeated acquisitions of a biometric feature may lead to different, but small variations in this data. When using a classical hash function (eg. SHA1, SHA2) or encryption function (eg. AES) on these different variations, a completely different result will be obtained.

In order to overcome this problem, mechanisms such as fuzzy extractor [47], biohashing [48] or cancelable biometrics [49] are applied to the biometric data, allowing the derivation of a fixed user-specific random vector. We refer to the literature for more details on these mechanisms. Therefore, when we mention biometric data of a user in the protocol, we mean the output which is generated by one of the above-mentioned mechanisms.

3.3 System setting

The main entities in the system are as follows.

- The user and its smartphone, which is provided with biometric feature extraction. The smartphone possesses a secure tamper resistant module (TRM) to store user-specific, temporary key material, and no long-term system security parameters.
- There are multiple servers, each offering a particular service.
- The central authority (CA) checks the identity of the users or servers, requesting to get access to the infrastructure. If positive, certificates are created for the user and server to register at the registration center. We assume a trustworthy CA as it is the basis of the whole system.
- The registration center (RC) is responsible for the registration of the servers and users, based on a valid identification performed by the CA. We assume an honest, but curious RC.

We assume that the required cryptographic functions and parameters are implemented in each entity participating in the scheme. We further assume that all smartphones of the users, servers, RC and CA are time synchronized, as timestamps are used to avoid replay attacks.

3.4 Notations

Table 1 summarizes the most important abbreviations and notations used in this paper.

4 Protocol EAAM

We distinguish five main phases in the protocol: registration with CA, registration with server, login phase, authentication and session key establishment, and renewal of user's password.

Table 1 Abbreviations and notations

U, U_d	User and user's device
TRM	Tamper resistant module in user's device
RC, S	Registration center and server
CA	Central authority
x	Public system parameter, the seed generating Chebyshev chaotic map
y	Secret parameter of RC
p	Public system parameter, large prime number
GK	Group key shared between RC and registered servers
$t_n = T_n(x)$	Chebyshev polynomial of degree n
$t_{nm} = T_n(T_m(x))$	Chebyshev polynomial of degree n with input $T_m(x)$
$(i, t_i = T_i(x))$	Secret and public key of the user
$(r, t_r = T_r(x))$	Secret and public key of the RC
$(s, t_s = T_s(x))$	Secret and public key of the server
ID_i, ID_S	Identity of the user U and server S
T_i	Timestamp at the registration time of U with CA
$cert_i$	Certificate on (ID_i, t_i, T_i) of U by CA
PW, Bio	Password and biometrics data of the user U
ϵ	Maximum allowed delay between submission and reception
$E_K(.), D_K(.)$	Symmetric key encryption and decryption with key K
$H(.)$	Hash function
$\|, \oplus$	Concatenation, bitwise XOR operation

4.1 Registration with CA

We need to distinguish actions performed by the RC, the user and its device, and the servers.

4.1.1 Actions to be performed by the RC

Let y be a secret parameter of the RC. We assume the RC has published the following system parameters:

- x as the seed for generating the Chebyshev chaotic map
- p as a large prime number, defining the modulo operation on the output of the Chebyshev chaotic map
- $t_r = T_r(x)$, the public key of the RC, together with a certificate $cert_r$ on (ID_r, t_r)
- $H(.)$, $E_K(.)$, $D_K(.)$, representing the hash algorithm, symmetric key encryption and decryption algorithm with secret shared key K
- ϵ is the predefined delay between submission and reception

4.1.2 Actions to be performed by the servers

The server generates its own secret key s and public key t_s . After approval from the CA, a certificate is generated by the CA. This certificate, together with its identity ID_S and public key t_s, is published in a publicly available repository.

The RC shares with all registered and validated servers a group key GK through a well-known multicast group key mechanism, like [50]. This key is regularly updated and in particular when a server is added or removed to the subscription list. Note that the previous versions of the group key, together with the period of their life time, are still stored at the servers until the expiration date.

4.1.3 Actions to be performed by the user and its device

Before the user can register, he/she first needs to generate its own key pair (i, t_i) and requests a certificate $cert_i$ on the combination of identity ID_i and public key t_i by the CA. These actions are detailed below.

- *Public-private key definition of the user*
  The user enters its identity ID_i, password PW and biometric data Bio into its smartphone. Based on a random variable rb stored in the memory of the phone, the private key $i = H(ID_i \| PW \oplus rb \| Bio)$ is computed. Next, the corresponding public key t_i is derived for it. The values ID_i, t_i are sent to the CA through a secure channel.
- *Response of CA*
  The CA checks the request by verifying the identity of the user. If positive, it validates the link between ID_i and its corresponding claimed public key t_i by generating a certificate $cert_i$ on $(H(ID_i, t_i, T_i))$, where T_i represents the timestamp at the moment of the registration request.
- *Storage on U_d*
  The device stores the received values $cert_i, t_i, T_i$, together with the parameter rb and $E_i = H(PW \oplus rb \| Bio \oplus rb \| ID_i)$. The parameters i and ID_i are stored in the TRM of the device.

4.2 Registration with RC

Using the information $(ID_i, t_i, T_i, cert_i)$, the user is now able to register with the RC without the need of a secure channel by performing the following operations. We assume that the public key of the RC can be consulted through secure publicly available repositories. The following steps are executed. The visualization of this process is depicted in Fig. 1.

- First, the device computes the value $F_i = H(i \| ID_i)$. Then, the user selects a random value $l \in \mathbb{R}$ and computes t_l. Next, also t_{lr} and t_{ir} are computed, using the public key t_r of the RC. Based on the current timestamp T_1, a dynamic secret k_1 with the RC can be derived, ie. $k_1 = H(t_{ir} \| T_1 \| ID_i)$. Finally, the key $t'_{lr} = H(t_{lr} \| T_1)$ is now used to encrypt the message containing this key k_1, the value F_i, and the

U, U_d		RC
$cert_i, t_i, T_i, E_i, rb$ i, ID_i		r, t_r, GK, y
$F_i = H(i \Vert ID_i)$ $l \leftarrow_R \mathbb{R}, t_l$ $t_{lr} = T_l(T_r(x))$ $t_{ir} = T_i(T_r(x))$ $k_1 = H(t_{ir} \Vert T_1 \Vert ID_i)$ $t'_{lr} = H(t_{lr} \Vert T_1)$ $C_1 = E_{t'_{lr}}(k_1 \Vert F_i \Vert ID_i \Vert t_i \Vert T_i \Vert cert_i \Vert t_r)$	$\xrightarrow{C_1 \Vert T_1 \Vert t_l}$	
		$T_2 - T_1 \leq \epsilon$ $t_{rl} = T_r(T_l(x))$ $t'_{rl} = H(t_{rl} \Vert T_1)$ $k_1 \Vert F_i \Vert ID_i \Vert t_i \Vert T_i \Vert cert_i \Vert t_r = D_{t'_{lr}}(C_1)$ $t_{ri} = T_r(T_i(x))$ Check $cert_i, k_1$ $A_i = H(ID_i \Vert GK \Vert y \Vert n)$ $B_i = A_i \oplus GK$ $C_i = H(A_i) \oplus H(F_i \Vert t_i \Vert ID_i)$ $k_2 = H(t_{rl} \Vert T_1 \Vert T_2 \Vert ID_i)$ $sig = sig_{RC}(H(T_2 \Vert GK \Vert B_i \Vert t_i))$ $C_2 = E_{k_2}(B_i \Vert C_i \Vert T_2 \Vert sig)$
	$\xleftarrow{C_2 \Vert T_2}$	
$T_3 - T_2 \leq \epsilon$ $k_2 = H(t_{lr} \Vert T_1 \Vert T_2 \Vert ID_i)$ $B_i \Vert C_i \Vert T_2 \Vert sig = D_{k_2}(C_2)$ Check T_2 Store B_i, C_i, T_2, sig		

Fig. 1 Steps and computations during the registration request of user with RC

credentials of the user and the RC, ie. $C_1 = E_{t'_{lr}}(k_1 \Vert F_i \Vert ID_i \Vert t_i \Vert T_i \Vert cert_i \Vert t_r)$. The message C_1, t_l, T_1 is sent to the RC.

– Upon arrival, the RC first checks the validity of the timestamp, if the current timestamp T_2 satisfies $T_2 - T_1 \leq \epsilon$. If so, the RC computes t_{ri}, t_{rl} and t'_{rl}. This last one is used to decrypt C_1. In order to authenticate the user with identity ID_i, the certificate $cert_i$ is verified and the value of $H(t_{ir} \Vert T_1 \Vert ID_i)$ is compared with k_1 from the decrypted message. If all checks are positive, then the RC computes the following parameters. Let n be the number of registrations performed by ID_i.

$$
\begin{aligned}
A_i &= H(ID_i \Vert GK \Vert y \Vert n) \\
B_i &= A_i \oplus GK \\
C_i &= H(A_i) \oplus H(F_i \Vert t_i \Vert ID_i) \\
k_2 &= H(t_{rl} \Vert T_1 \Vert T_2 \Vert ID_i) \\
sig &= sig_{RC}(H(T_2 \Vert GK \Vert B_i \Vert t_i)) \\
C_2 &= E_{k_2}(B_i \Vert C_i \Vert T_2 \Vert s)
\end{aligned}
$$

Here, $sig_{RC}(H(T_2 \Vert GK \Vert B_i \Vert t_i)))$ denotes the signature of RC over the message $M = H(T_2 \Vert GK \Vert B_i \Vert t_i))$. The message C_2, T_2 is sent to the user.

– After a positive validation of the timestamp, the user computes k_2 and encrypts C_2. If T_2 from this message corresponds with the submitted T_2, the user stores the parameters B_i, C_i, T_2, sig, besides the already stored values $E_i, rb, cert_i, t_i, T_i$ and i, ID_i in TRM. Note that the user is not able to verify the signature, as it does not know the group key GK. Instead, the signature verification will be performed later by the servers.

We want to note that each parameter in the derivation of RC has a specific goal, as will be later shown in more detail in the security analysis. To be short, A_i is meant to be only constructible by the RC (knowledge of y) and C_i by the user (knowledge of F_i and thus i, ID_i). The user does not know A_i, but given B_i and $H(A_i)$ (from C_i), only the servers knowing GK can verify this link and thus identify a valid user.

4.3 User login

The user takes its smartphone and inputs its identity ID_i, password PW_i, and biometric data Bio. If $H(PW \oplus rb \| Bio \oplus rb \| ID_i)$ corresponds to the stored E_i, the process can continue. Then, the smartphone computes $F_i = H(i \| ID_i)$. It looks up the public key t_s of the server S in a publicly available secured repository. Next, the device picks up a random value f and computes t_f and t_{fs}. Now, the following computations are performed in order to derive the user login request to be transmitted, with T_3 the current timestamp.

$$\begin{aligned} H(A_i) &= C_i \oplus H(F_i \| t_i \| ID_i) \\ CID_i &= B_i \oplus H(t_{fs} \| T_3) \\ k_3 &= H(T_3 \| t_{fs}) \oplus H(A_i) \\ C_3 &= E_{k_3}(sig \| t_i \| T_2) \end{aligned}$$

Consequently, the message t_f, T_3, CID_i, C_3 will be transmitted to S.

4.4 Authentication phase and key agreement

Upon arrival of the message at T_4 from the user login phase, the server first checks if the timestamp of the user is valid, meaning $T_4 - T_3 \leq \epsilon$. If so, t_{sf} will be computed and the following operations are performed.

$$\begin{aligned} B_i &= CID_i \oplus H(t_{sf} \| T_3) \\ A_i &= B_i \oplus GK \\ k_3 &= H(T_3 \| t_{fs}) \oplus H(A_i) \\ sig \| t_i \| T_2 &= D_{k_3}(C_3) \\ M &= H(T_2 \| GK \| B_i \| t_i) \end{aligned}$$

If $sig_{RC}(M)$ corresponds with sig of C_3, the server is guaranteed that the credentials are derived by the RC and not by a potential malicious server. However, note that it

U, U_d		S
$t_s, B_i, C_i, T_2, sig_{,i}, T_i, cert_i, E_i, rb$		s, t_s, GK, t_r
ID_i, i		
$ID_i, PW, Bio \rightarrow$		
$E_i? = H(PW \oplus rb \Vert Bio \oplus rb \Vert ID_i)$		
$F_i = H(i \Vert ID_i)$		
$f \leftarrow_R \mathbb{R}, t_f$		
$t_{fs} = T_f(T_s(x))$		
$H(A_i) = C_i \oplus H(F_i \Vert t_i \Vert ID_i)$		
$CID_i = B_i \oplus H(t_{fs} \Vert T_3)$		
$k_3 = H(T_3 \Vert t_{fs}) \oplus H(A_i)$		
$C_3 = E_{k_3}(sig \Vert t_i \Vert T_2)$		
	$\xrightarrow{C_3 \Vert CID_i \Vert t_f \Vert T_3}$	
		$T_2 - T_1 \leq \epsilon$
		$t_{sf} = T_s(T_f(x))$
		$B_i = CID_i \oplus H(t_{sf} \Vert T_3)$
		$A_i = B_i \oplus GK$
		$k_3 = H(T_3 \Vert t_{fs}) \oplus H(A_i)$
		$sig \Vert t_i \Vert T_2 = D_{k_3}(C_3)$
		$M = H(T_2 \Vert GK \Vert B_i \Vert t_i)$
		Check $sig = sig_{RC}(M)$
		$b \leftarrow_R \mathbb{R}, t_{bi} = T_b(T_i(x)), t_{bf} = T_b(T_f(x))$
		$GK = H(t_{bf} \Vert t_{bi} \Vert t_{sf} \Vert T_3 \Vert T_4)$
		$C_4 = H(SK \Vert t_{bf} \Vert t_{bi} \Vert t_{sf} \Vert T_3 \Vert T_4)$
	$\xleftarrow{C_4 \Vert t_b \Vert T_4}$	
$T_5 - T_4 \leq \epsilon$		
$t_{ib} = T_i(T_b(x)), t_{fb} = T_f(T_b(x))$		
$SK = H(t_{bf} \Vert t_{bi} \Vert t_{fs} \Vert T_3 \Vert T_4)$		
Check $C_4 = H(SK \Vert t_{bf} \Vert t_{ib} \Vert t_{fs} \Vert T_3 \Vert T_4)$		

Fig. 2 Steps and computations during the user login and authentication and key agreement phase

might still be possible that a malicious server has stored the signatures of previous requests of its users and tries to abuse it in the name of a user. Therefore, it is essential that the SK should also exploit the usage of the public key t_i. This symmetric shared key equals $SK = H(t_{bf} \Vert t_{bi} \Vert t_{sf} \Vert T_3 \Vert T_4)$, where b is a randomly chosen parameter and t_b the corresponding public variant of it. Finally, $C_4 = H(SK \Vert t_{bf} \Vert t_{bi} \Vert t_{sf} \Vert T_3 \Vert T_4)$ is computed and C_4, t_b, T_4 is sent to the user. Note that it is possible to add some restrictions, being that the delay between user login T_3 and reception of credentials T_2 is in a predefined range.

With this information, the user's device first verifies the timestamp T_4. Next, U_d computes t_{ib}, t_{fb}, and thus also SK. Finally, if the hash value of $H(SK \Vert t_{fb} \Vert t_{ib} \Vert t_{fs} \Vert T_3 \Vert T_4)$ corresponds with the received C_4, then user and server have a common shared secret key SK. tamper The different steps from the user login and the authentication and key agreement phase are visualized in Fig. 2.

4.5 Password change phase

If the user wants to change its password, it can complete the procedure without involvement of the RC. The user first logs in with identity and password. The device checks the entered information with E_i. If the user is authentic, the device prompts the user for a new password PW^* and computes

$$rb^* = PW \oplus PW^* \oplus rb$$

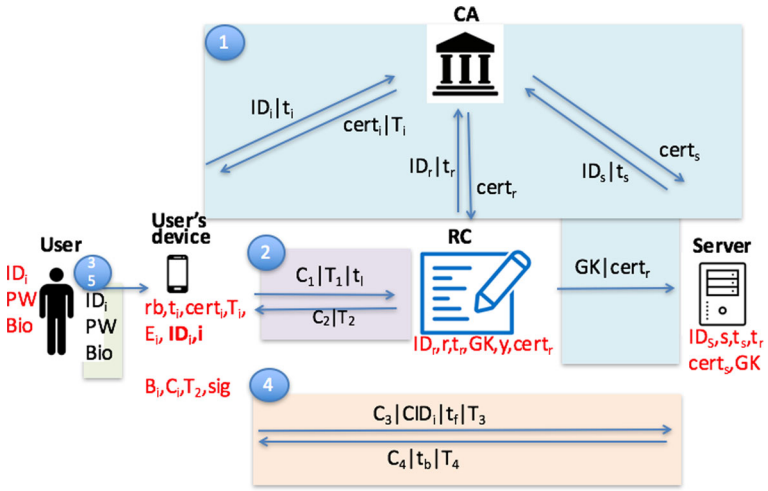

Fig. 3 Communication flow of the five phases—(1) registration phase with CA, (2) registration phase of user with RC, (3) user login, (4) authentication and key agreement phase, (5) password change phase. The stored key material at each entity is denoted in red. The key material to be stored in the tamper resistant part of the memory of the user's device is denoted in bold (color figure online)

$$
\begin{aligned}
i^* &= i = H(ID_i \| PW^* \oplus rb^* \| Bio) \\
t_i^* &= t_i = T_{i^*}(x) \\
E_i^* &= H(PW^* \oplus rb^* \| Bio \oplus rb^* \| ID_i) \\
F_i^* &= F_i = H(i^* \| ID_i) \\
C_i^* &= C_i = H(A_i) \oplus H(F_i^* \| t_i^* \| ID_i)
\end{aligned}
$$

Finally, the values of E_i, rb are updated by E_i^*, rb^*. The other values remain unchanged.

To conclude, Fig. 3 depicts the flow of the different phases, (1) registration phase with CA, (2) registration phase of user with RC, (3) user login, (4) authentication and key agreement phase, (5) password change phase. The key material stored in the different entities is denoted in red.

5 Security analysis

We focus the security analysis on the user login, authentication and key agreement phase, which is the core of the system.

5.1 Mutual authentication

Mutual authentication means that both entities authenticate each other in the protocol. We distinguish the authentication of S with U and the other way around.

- *S with U* Only the specific S can derive B_i from CID_i, by the computation of t_{fs} using its private key s. In addition, it can check the validity of the request by verifying the relation with $A_i = B_i \oplus GK$ and checking the validity of the key k_2 such that the last part of the decryption of C_3 corresponds with the public key included in the signature sig. In order to check that the request is really validated by the RC and not a malicious server knowing the GK, the signature of the RC on the identity (public key t_i) and the timestamp is required to be included in the message and to be verified. As a malicious server can also store the signatures of the previous requests of its users, the symmetric shared key between S and U also includes t_{bi}, which requires the knowledge of the private key i or the randomly defined parameter b by S. Consequently, the server is guaranteed that only a valid user is able to derive the symmetric shared key SK.
- *U with S* As U makes use of the public key of S, he is sure that only the entity knowing the corresponding private key is able to derive the required parameters, due to the DL problem. Moreover, the message also implicitly includes GK, in both the signature and the definition of A_i.

5.2 Impersonation attacks and man-in-the-middle attacks

This attack implies that an attacker listening to the messages between the intended partners would intercept the communication and would take over the role as legitimate participant. Due to the intensive usage of the public keys of the server and the user on the one hand and the key material derived by the RC from the other hand, it is impossible for an outsider to derive useful information from the messages or to construct legitimate alternatives.

5.3 Insider attacks

Insider attacks are attacks where the involved parties try to abuse the system by sending fake but legitimate requests, allowing the derivation of a secret shared key. In this protocol, we need to distinguish according to the 2 roles of user and server.

A malicious server is able to send a valid user login request, using the stored signature of a previous user request. However, he is not able to derive a symmetric shared secret key as this key involves t_{bi}, which requires the knowledge of the private key of the user. Therefore, a malicious server is not able to mimic a user. He also cannot derive any useful information of the messages from the login phase, since both V_1 and CID_i integrate the parameter t_{fs}, only derivable by the server S.

Finally, also a malicious user is not able to generate fake credentials, as he is not aware of the GK or the private key of RC to make a valid signature.

5.4 Replay attacks

A replay attack can be launched when an attacker replays the original message or parts of it at a later moment in time in order to betray or impersonate one of the involved parties. The messages sent in our protocol are $C_3 \| V_1 \| CID_i \| t_f \| T_3$ and $C_4 \| t_b \| T_4$.

Replay attacks on the first message are impossible due to the usage of the timestamp T_3, which is included in the other parameters $C_3 \| V_1 \| CID_i$. Similar for the second message, the timestamp T_4 is included in the computation of C_4.

5.5 Known key attacks

These type of attacks allow the derivation of the private keys of S or U from the session keys. Since $SK = H(t_{bf} \| t_{bi} \| t_{fs} \| T_3 \| T_4)$ is the result of the hash of t_{bf}, t_{bi}, t_{fs}, it is impossible to derive information on i or s, due to the one way property of the hash function and in addition due to the CDH problem.

5.6 Perfect forward secrecy

Even if the secret keys of the user or the server are leaked, the previously generated secret session keys remain secret. This follows from the fact that t_{fb} is impossible to derive, without knowledge of b and f and these temporarily generated values are erased from the memories of U_d and S after usage.

5.7 Resistance against password guessing/stolen smartcard attacks

In this type of attacks, an attacker might steal the smartphone and use all information available on the phone (e.g. by using power analysis techniques [51,52]) together with all public messages sent over the public channel. The proposed protocol is secure against such attack as it is a three-factor-based protocol, meaning that the knowledge of the password, the possession of the device, and the inheritance of the biometric data should be present with the user for a successful login. Consequently, an attacker in the possession of U_d, but without knowledge of the password of the biometric data, is not able to activate the phone.

Even if a malicious person gets access to the device and succeeds in deriving the stored parameters $E_i, B_i, C_i, t_i, T_2, s, rb, cert_i$. He still does not know the variables i, ID_i in the TRM of the phone. These parameters are required to compute t_{ir} and thus k_1 in the registration request to the RC. Also i and ID_i are required in the user login and key agreement with the server to compute F_i, and thus $H(A_i)$.

5.8 Prone to curious RC

A curious but honest RC is one that follows the required procedures but wants in the end to derive useful information of this users for its own purpose. The session key in EAAM is only derivable by the involved user and server, as explained above. In particular, the RC is not able to derive the key SK, since SK includes the parameters t_{bf}, t_{bi}, t_{fs}, which can due to the CDH problem only be generated with the knowledge of either the private keys or the randomly derived parameters of the participants.

5.9 Anonymity

Anonymity provides privacy to the users. Anonymity to any outsider is clearly obtained as none of the transmitted messages contain useful information which can link to the identity of the user. Also unlinkability is established due to the usage of a dynamic identity CID_i.

Anonymity of the identity of the user by the server is obtained in the case the server is not able to link the public key t_i to a specific server. In case the server needs to reveal the identity of the user, the user sends its identity ID_i instead of its public key t_i in C_3 during the user login request. In this case, the server needs to get access to a publicly secure directory containing the tuples identity and their corresponding public key.

Note that in none of the cases unlinkability of the user by the server can be obtained, without constantly changing the certificate of the user.

5.10 Access bounded in time

Due to the fact that the certificate includes a timestamp T_2, the server has knowledge on the existence date of the certificate and make a decision of the validity on it based on the delay between the request time T_3 and T_2.

5.11 Summary

We now compare our scheme with the three most efficient systems [36–38] in the literature using Chebyshev polynomials, which also only require 2 communication rounds in the authentication request and satisfy most of the required security properties.

None of them satisfy the property of resistance against a curious RC. In addition, [36] is not resistant to insider attacks with malicious servers, and [37] does not satisfy unlinkability. Both [37,38] work with a certificate revocation list (CRL) for bounding the access to privileged non revoked users, while [36] does not consider this aspect. In EAAM, the access of users is only limited in time (depending on the validity period of the signature of RC determinable by each individual server). The advantage of this approach, compared to the usage of a CRL on the RC's end, is that in this case no permanent communication between RC and S is required and a differentiated approach for each server is possible. Moreover, in most practical scenarios, the certificate is valid upon a certain amount of time. Finally, the last difference is that [36] is a two-factor-based authentication protocols, while EAAM and [37,38] are all three-factor based, including biometric information.

To conclude, Table 2 summarizes the comparison of the most challenging security features and shows the difference between our protocol EAAM and [36–38].

6 Performance analysis

We focus on the operations and communication requirements from the side of the user, as it is most critical from computational point of view. Moreover, we limit the

Table 2 Comparison of security features

Protocol	Anonymity	No insider attacks	Perfect forward secrecy	Unlinkability	Curious RC	No secure channel (RC-U)
EAAM	+	+	+	+	+	+
[36]	+	−	+	+	−	−
[37]	+	+	−	−	−	−
[38]	+	+	+	+	−	−

Table 3 Comparison of the communication requirements

Protocol	Number of transmitted bits	Number of received bits
EAAM	608	224
[36]	384	224
[37]	640	640
[38]	384	224

discussion to the user login and authentication request phase as the registration phase is only performed once. Moreover, our system does not require a secure channel between the user and the RC, while the others need such channel to share secret parameters.

6.1 Communication requirements

We assume, similar to [36–38], that the number of bits for the identity and timestamps equals to 32. The number of bits as a result of the hash operation is 160 bits (eg. SHA 1) and the result of encryption and decryption is a multiple of 128 bits (eg. AES). The comparison of the number of transmitted and received bits is summarized in Table 3. We can conclude that our protocol requires an additional 224 bits to send a message, compared to the best protocols [36,38]. This follows from the fact that we need to include the signature of the CA. Note that we have a very compressed signature as the message to be signed is already derived from the previous parameters. Only the two auxiliary variables for the signature generation need to be included in sig. The number of received bits is equal to the best performing systems in literature.

6.2 Computational requirements

The operations to be considered in the comparison are the hash operation, bio hash operation, symmetric encryption and decryption algorithm, and the Chebyshev polynomial evaluation. The XOR operation is negligible, compared to these operations. In [53], the authors state that the time for executing a biohashing operation is similar to the time required for an elliptic curve multiplication. Also the time to evaluate a Chebyshev polynomial can be approximated by the time corresponding with the exe-

Table 4 Computation time of the operations [55]

Operation	Description	Time (in ms)
T_H	Time for SHA 1 Hash operation	0.1
T_{BH}	Time for bio hash operation	1.7
T_E	Time for encryption or decryption with AES 128	0.1
T_C	Time for Chebyshev polynomial map evaluation	0.12

Table 5 Comparison of the performance

Protocol	Computation cost	Total time (in ms)
EAAM	$7T_H + 1T_{BH} + 1T_E + 4T_C$	3.12
[36]	$5T_H + 3T_C$	0.96
[37]	$11T_H + 1T_{BH} + 2T_E + 4T_C$	3.7
[38]	$6T_H + 1T_{BH} + 4T_C$	2.9

cution of a hashing operation [54]. Table 4 shows the execution time for each of these operations as derived in [55]. The measurements are performed on a 2260 MHz ARM device, corresponding with a smartphone with Android OS (Android 4.4 KitKat), containing a 32-bit processor, ARMv7 Quadcore. The timing values are measured using the system clock. The test applications are written in the JAVA programming language by using the Android Software Development Kit (SDK) and the available crypto APIs. The resulting performance value is obtained after taking the average of 10 iterations.

As a result, Table 5 compares the performance of EAAM with the systems [36–38], both in terms of computation cost and in total time. Note that all these schemes consist of two communication rounds in the authentication request of the user to the server and are based on Chebyshev polynomials.

We can conclude that the most efficient protocol [36] is almost double efficient as the others. This follows from the first place since it is a two-factor-based protocol, and thus not requires a bio hash operation (corresponding to the most compute intensive operation). Next, the protocol of [36] also satisfies a clear weakness in the sense that it is not secure against insider attacks of malicious servers and can in fact from a security point of view be seen as only a single server system. The main difference between our protocol and [38] is an additional encryption and hash for the derivation of the encryption key, which is required to send the certificate of the CA. As the protocols require less than 350 ms in general, they can be considered as real time from a user's point of view [55].

7 Conclusion

This paper presents an efficient anonymous authentication protocol in multiple server communication networks, called *EAAM*. In the EAAM, the user required three factors (e.g., knowledge of password, possession of biometrics and smartphone) for access-

ing multiple servers. The main novelty of the protocol is that it provides resistance against a honest-but-curious RC. Especially with respect to the privacy concerns of the user, it is very important to know that no big brother in the system exists which is able to collect or analyse all information of the different users to be exploited for different purposes without awareness of the user. Moreover, EAAM also offers very elegantly a differentiated validity period of its access permission delivered by the RC for the different servers and does not work by means of a CRL requiring additional communication between RC and S during the authentication request.

We have discussed the security strengths of the system and have shown that the protocol is very competitive with respect to computational and communication requirements, compared to the state of the art and taken into account its additional security features.

References

1. Hwang M, Li L (2000) A new remote user authentications scheme using smart cards. IEEE Trans Consum Electron 46(1):28–30
2. He D, Zeadally S, Wang H, Liu Q (2017) Lightweight data aggregation scheme against internal attackers in smart grid using elliptic curve cryptography. Wirel Commun Mob Comput 2017:11
3. Li L, Peng H, Kurths J, Yang Y, Schellnhuber HJ (2014) Chaos-order transition in foraging behavior of ants. PNAS 111(23):8392–8397
4. Tashi J (2014) J., Comparative analysis of smart card authentication schemes. IOSR J Comput Eng 16(1):91–97
5. Huang X, Xiang Y, Chonka A, Zhou J, Deng RH (2011) A generic framework for three-factor authentication: preserving security and privacy in distributed systems. IEEE Trans Parallel Distrib Syst 22(8):1390–1397
6. Dodis Y, Reyzin L, Smith A (2004) Fuzzy extractors: how to generate strong keys from biometrics and other noisy data. In: Proceedings of EUROCRYPT, pp 523–540
7. Banerjee S, Dutta MP, Bhunia CT (2015) An Improved smart card based anonymous multi-server remote user authentication scheme. Int J Smart Home 9(5):11–22
8. Baruah KCH, Banerjee S, Dutta MP, Bhunia CT (2015) An improved biometric-based multi-server authentication scheme using smart card. Int J Secur Appl 9(1):397–408
9. Li CT, Hwang MS (2010) An efficient biometrics based remote user authentication scheme using smart cards. J Netw Comput Appl 33(1):1–5
10. Chuang MC, Chen MC (2014) An anonymous multi-server authenticated key agreement scheme based on trust computing using smart cards and biometrics. Expert Syst Appl 41(4):1411–1418
11. Mishra D, Das AK, Mukhopadhyay S (2014) A secure user anonymity-preserving biometric-based multi-server authenticated key agreement scheme using smart cards. Expert Syst Appl 41(18):8129–8143
12. Das AK (2011) Analysis and improvement on an efficient biometric based remote user authentication scheme using smart cards. IET Inf Secur 5(3):145–151
13. An Y (2012) Security analysis and enhancements of an effective biometric-based remote user authentication scheme using smart cards. J Biomed Biotechnol 2012:6
14. Khan MK, Kumari S (2013) An improved biometrics-based remote authentication scheme with user anonymity. J Biomed Biotechnol 9
15. Wen F, Susilo W, Yang G (2015) Analysis and improvement on a biometric-based user authentication scheme using smart cards. Wireless Pers Commun 80:1747–1760
16. Braeken A, Porambage P (2015) Efficient anonym smart card based authentication scheme for multi-server. Architecture 9(9):177–184
17. Pippal RS, Wu S (2013) Robust smart card authentication scheme for multi-server architecture. Wireless Pers Commun 72(1):729–745
18. Wei J, Liu W, Hu X (2014) Cryptanalysis and improvement of a robust smart card authentication scheme for multi-server architecture. Wireless Pers Commun 77(1):2255–2269

19. Lin H, Wen F, Du C (2015) A novel and anonymous key agreement multi-server architecture. J Comput Inf Syst 11(8):3011–3018
20. Yoon E, Yoo K (2013) Robust biometrics-based multi-server authentication with key agreement scheme for smart cards on elliptic curve cryptosystem. J Supercomput 63(1):235–255
21. Kim H, Jeon W, Lee K, Lee Y, Won D (2012) Cryptanalysis and improvement of a biometrics-based multi-server authentication with key agreement scheme. In: Proceedings of 12th International Conference on Computational Science and its Applications (ICCSA 2012), Salvador de Bahia, pp 391–406
22. He D, Wang D (2015) Robust biometrics-based authentication scheme for multiserver environment. IEEE Syst J 9(3):816–823
23. Odelu V, Das AK, Goswami A (2015) A secure biometrics-based multiserver authentication protocol using smart cards. IEEE Trans Inf Forensics Secur 10(9):1953–1966
24. Jiang Q, Khan MK, Lu X, Ma J, He D (2016) A privacy preserving three-factor authentication protocol for e-health clouds. J Supercomput 72:1–24
25. Jiang Q, Ma J, Li G, Yang L (2014) An efficient ticket based authentication protocol with unlinkability for wireless access networks. Wireless Pers Commun 77(2):1489–1506
26. Liao YP, Hsiao CM (2013) A novel multi-server remote user authentication scheme using selfcertified public keys for mobile clients. Future Gener Comput Syst 29(3):886–900
27. Amin R, Biswas GP (2016) Design and analysis of bilinear pairing based mutual authentication and key agreement protocol usable in multi-server environment. Wirel Pers Commun 84(1):439–462
28. Guo C, Chang C-C (2013) Chaotic maps-based password authenticated key agreement using smart cards. Commun Nonlinear Sci Numer Simul 18(6):1433–1440
29. Hao X, Wang J, Yang Q, Yan X, Li P (2013) A chaotic map based authentication scheme for telecare medicine information systems. J Med Syst 37(2):1–7
30. Jiang Q, Ma J, Lu X, Tian Y (2014) Robust chaotic map based authentication and key agreement scheme with strong anonymity for telecare medicine information systems. J Med Syst 38(2):1–8
31. Lee CC, Chen CC, Wu CY, Huang S-Y (2012) An extended chaotic maps-based key agreement protocol with user anonymity. Nonlinear Dyn 69(1–2):79–87
32. Lee CC, Hsu CW (2013) A secure biometric-based remote user authentication with key agreement scheme using extended chaotic maps. Nonlinear Dyn 71(1–2):201–211
33. Islam SKH (2014) Provably secure dynamic identity-based three-factor password authentication scheme using extended chaotic maps. Nonlinear Dyn 78(3):2261–2276
34. Khan MK, Zhang J, Wang X (2008) Chaotic hash-based fingerprint biometric remote user authentication scheme on mobile devices. Chaos Solitons Fractals 35(3):519–524
35. He D, Chen Y, Chen J (2012) Cryptanalysis and improvement of an extended chaotic maps-based key agreement protocol. Nonlinear Dyn 69(3):1149–1157
36. Lee CC, Lou DC, Li CT (2013) An extended chaotic maps based protocol with key agreement for multiserver environments. Nonlinear Dyn 76(1):853–866
37. Chatterjee S, Roy S, Das AK, Chattopadhyay S, Kumar N, Vasilakos AV (2016) Secure biometric-based authentication scheme using Chebyshev chaotic map for multi-server environment. In: IEEE Transactions on Dependable and Secure Computing
38. Irshad A, Sher M, Chaudhary SA, Naqvi H, Farash MS (2016) An efficient and anonymous multi-server authenticated key agreement based on chaotic map without engaging Registration Centre. J Supercomput 72(4):1623–1644
39. Hsieh WB, Leu JS (2014) An anonymous mobile user authentication protocol using self-certified public keys based on multi-server architectures. J Supercomput 70(1):133–148
40. Zhao D, Peng H, Li YYS (2013) An efficient dynamic ID based remote user authentication scheme using self-certified public keys for multi-server environment. CoRR abs/1305.6350
41. Tseng YM, Wu TY, Wu J (2008) A pairing-based user authentication scheme for wireless clients with smart card. Informatics 19(2):285–302
42. Geng J, Zhang L (2008) A dynamic id-based user authentication and key agreement scheme for multi-server environment using bilinear pairings. In: Workshop on Power Electronics and Intelligent Transportation System (PEITS 2008), Guangzhou, pp 33–37
43. Bergamo P, Arco P, Santis A, Kocarev L (2005) Security of public key cryptosystems based on Chebyshev polynomials. IEEE Trans Circ Syst 52:1382–1393
44. Kocarev L, Lian S (2011) Chaos-based cryptography: theory, algorithms and applications. Springer, Berlin ISBN 978-3-642-20542-2

45. Zhang L (2008) Cryptanalysis of the public key encryption based on multiple chaotic systems. Chaos Solitons Fractals 37(3):669–674
46. Chain K, Kuo WC (2013) A new digital signature scheme based on chaotic maps. Nonlinear Dyn 74(4):1003–1012
47. Dodis Y, Reyzin L, Smith A (2004) Fuzzy extractors: how to generate strong keys from biometrics and other noisy data. FSIAM J Comput 38(1):97–139
48. Jin A, Ling D, Goh A (2004) Biohashing: two factor authentication featuring fingerprint data and tokenised random number. Pattern Recogn 37(11):2245–2255
49. Ratha NK, Connell JH, Bolle RM (2001) Enhancing security and privacy in biometrics-based authentication systems. IBM Syst J 40(3):614–634
50. Porambage P, Braeken A, Schmitt C, Gurtov AV, Ylianttila M, Stiller B (2015) Group key establishment for enabling secure multicast communication in wireless sensor networks deployed for IoT applications. IEEE Access 3:1503–1511
51. Messerges TS, Dabbish EA, Sloan RH (2002) Examining smart-card security under the threat of power analysis attacks. IEEE Trans Comput 51(5):541–552
52. Kocher P, Jaffe J, Jun B (1999) Differential power analysis. In: Proceedings of Advances in Cryptology—CRYPTO99, LNCS, vol 1666. Springer, Santa Barbara, pp 388–397
53. He D, Kumar N, Lee JH, Sherratt RS (2014) Enhanced three-factor security protocol for consumer USB mass storage devices. IEEE Trans Consum Electron 60(1):30–37
54. Lee TF (2015) Provably secure anonymous single-sign-on authentication mechanisms using extended Chebyshev chaotic maps for distributed computer networks. IEEE Syst J PP(99):1–8
55. Malina HJ, Fujdiak R, Hosek J (2016) On perspective of security and privacy-preserving solutions in the internet of things. Comput Netw 102(19):83–95