PLOS ONE

RESEARCH ARTICLE

An Enhanced Biometric Based Authentication with Key-Agreement Protocol for Multi-Server Architecture Based on Elliptic Curve Cryptography

Alavalapati Goutham Reddy[1], Ashok Kumar Das[2], Vanga Odelu[3], Kee-Young Yoo[1]*

1 School of Computer Science and Engineering, Kyungpook National University, Daegu, Korea, **2** Center for Security, Theory and Algorithmic Research, International Institute of Information Technology, Hyderabad, India, **3** Department of mathematics, Indian Institute of Technology, Kharagpur, India

* yook@knu.ac.kr

Citation: Reddy AG, Das AK, Odelu V, Yoo K-Y (2016) An Enhanced Biometric Based Authentication with Key-Agreement Protocol for Multi-Server Architecture Based on Elliptic Curve Cryptography. PLoS ONE 11(5): e0154308. doi:10.1371/journal.pone.0154308

Editor: Wen-Bo Du, Beihang University, CHINA

Received: February 16, 2016

Accepted: April 11, 2016

Published: May 10, 2016

Data Availability Statement: Data are available at FigShare. DOI: 10.6084/m9.figshare.3204307; Link: https://figshare.com/s/1601551aea604ca5cec2.

Funding: This work was supported by the BK21 Plus project (SW Human Resource Development Program for Supporting Smart Life) funded by the Ministry of Education, School of Computer Science and Engineering, Kyungpook National University, Korea (21A20131600005), and Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) [No. 10041145, Self-Organized Software platform (SoSp) for Welfare Devices].

Abstract

Biometric based authentication protocols for multi-server architectures have gained momentum in recent times due to advancements in wireless technologies and associated constraints. Lu et al. recently proposed a robust biometric based authentication with key agreement protocol for a multi-server environment using smart cards. They claimed that their protocol is efficient and resistant to prominent security attacks. The careful investigation of this paper proves that Lu et al.'s protocol does not provide user anonymity, perfect forward secrecy and is susceptible to server and user impersonation attacks, man-in-middle attacks and clock synchronization problems. In addition, this paper proposes an enhanced biometric based authentication with key-agreement protocol for multi-server architecture based on elliptic curve cryptography using smartcards. We proved that the proposed protocol achieves mutual authentication using Burrows-Abadi-Needham (BAN) logic. The formal security of the proposed protocol is verified using the AVISPA (Automated Validation of Internet Security Protocols and Applications) tool to show that our protocol can withstand active and passive attacks. The formal and informal security analyses and performance analysis demonstrates that the proposed protocol is robust and efficient compared to Lu et al.'s protocol and existing similar protocols.

Introduction

The swift expansion of communication technologies and handheld devices have necessitated the authentication of every remote user. Authentication process verifies the legitimacy of each user and offers the access to network resources. Password, smartcard and biometrics based authentication are the few common technologies deployed until today. The first remote user password based authentication method was proposed by Lamport [1] in 1981 for communication over insecure channels. However, password based authentication methods are elusive and

Competing Interests: The authors have declared that no competing interests exist.

prone to guessing attacks. Thus, the password with smartcard based methods have come into sight. Conversely, research has shown that password with smartcard based authentication methods are still prone to numerous attacks when the smartcard is stolen. The ascribed limitations of password and smartcard based authentication methods have imposed to install additional security methods such as biometrics. Biometric keys such as palm print, iris, finger print, face and so on are unique and secure. Biometrics with smartcards or passwords makes the authentication process very robust due to the following features [2] [3]:

- Biometric keys are non-forgeable and non-distributable.
- Biometric keys cannot be lost nor forgotten.
- It is extremely difficult to guess biometric keys unlike passwords.
- Breaking someone's biometrics is extremely difficult.

Few authentication technologies have used smartcards or biometrics or the both along with passwords [4–23]. Earlier authentication methods were limited to single-server architecture. This architecture is not adequate when the number of users with varied interests and open networks keep increasing. On the other hand, users are required to register at every server in order to avail the services, which is extremely tedious and adds the cost enormously. As a scalable solution, multi-server architecture has been introduced, where the users can register only once at the registration server and avail the services of all associated application servers. Several authors have suggested various authentication protocols for multi-server architecture during the past decade [24–46].

In 2009, Liao et al. [38] proposed a secure dynamic ID based remote user authentication protocol for multi-server environment. In the same year, Hsiang et al. [28] presented that Liao & Wang's protocol is prone to server and registration center spoofing attacks, insider attacks and masquerade attacks. Furthermore, they proposed an improved dynamic identity based mutual authentication without verification tables. In 2011, Sood et al. [42] proved that Hsiang et al.'s protocol is also vulnerable to impersonation attacks, stolen smart card attacks and replay attacks. In addition, they improved the weaknesses of Hsiang et al.'s protocol and proposed a protocol with different levels of trust between two-servers. In 2012, Li et al. [35] found that Sood et al.'s protocol is susceptible to impersonation attacks, stolen smart card attacks and leak-of-verifier attacks. Then, they proposed an efficient dynamic identity based authentication protocol with smart cards and claimed that it overcomes all aforementioned drawbacks. However, in 2014, Xue et al. [45] proved that Li et al.'s protocol still cannot resist forgery attacks, eavesdropping attacks, denial-of-service attacks and so on. They even put forward a lightweight dynamic pseudonym identity based authentication and key agreement protocol without verification tables for multi-server architecture. In the same year, Chuang et al. [25] proposed an anonymous multi-server authenticated key agreement protocol based on trust computing using smartcards and biometrics. Their protocol is light-weight and provides multi-server authentication with user anonymity. Later on, Mishra et al. [2] in 2014 and Lin et al. [39] in 2015 pointed out several drawbacks of Chuang et al.'s protocol and proposed a secure anonymous three factor authentication protocol. In 2015, Lu et al. [40] proved that Mishra et al.'s protocol was too vulnerable to replay attacks and contains an insecure password changing phase. They proposed a robust biometric based authentication protocol for multi-server architecture.

Contributions of the paper

Achieving several security properties while maintaining the best performance is essential for any user authentication protocol. Several recent proposed protocols fail to satisfy security and

performance properties. One of such protocols is Lu et al.'s robust biometric based authentication protocol for multi-server architecture. This paper's keen analysis demonstrates the weaknesses of Lu et al.'s protocol such as lack of user anonymity, prone to server and user impersonation attacks, man-in-middle attacks, no perfect forward secrecy and clock synchronization problems. In addition, this paper proposes an enhanced biometric based remote user authentication with key agreement protocol for multi-server architecture without user verification tables. The proposed protocol is perfectly suitable for real time applications as it accomplishes simple elliptic curve cryptography operations, one-way hash functions, concatenation operations and exclusive-OR operations. The proposed protocol is not only light-weight but also achieves all the eminent security properties such as user anonymity, mutual authentication, no verification tables, perfect forward secrecy and resistance to numerous attacks. We proved that the proposed protocol can achieve mutual authentication using BAN logic [47] and the formal security of the proposed protocol is verified using the widely accepted AVISPA tool [48] to ensure the resistance to active and passive attacks. The security and performance analysis sections demonstrates that the proposed protocol is more robust and efficient than Lu et al.'s protocol and other existing protocols.

Organization of the paper

The remainder of the paper is organized as follows: Section 2 shows the preliminaries used in this paper. Section 3 provides the review of Lu et al.'s protocol. Section 4 crypt analyses Lu et al.'s protocol. Section 5 presents the proposed protocol. Section 6 portrays formal security analysis using BAN logic and informal security analysis of the proposed protocol in detail. In Section 7, the simulation for the formal security verification of the proposed protocol using the AVISPA tool shows that the proposed protocol is secure. Section 8 affords performance analysis and comparison with the related protocols. At last, Section 9 concludes the paper.

Review of Lu et al.'s Protocol

This section provides an overview of Lu et al.'s [40] biometrics based authentication with key-agreement protocol for multi-server architecture using smartcards. Lu et al.'s protocol comprises three participants, user (U_i), authorized server (S_j), registration center (RC) and four phases, registration phase, login phase, authentication phase, and password change phase. RC initializes the system by sharing the chosen secret key PSK and random number x with S_j via a secure channel. The various notations used in Lu et al.'s protocol are listed in Table 1.

Table 1. Notations of Lu et al.'s protocol.

U_i	An user
S_j	Authorized server
RC	Registration center
ID_i	Identity of U_i
PW_i	Password of U_i
BIO_i	Biometrics of U_i
x, y	Private keys of RC and U_i
PSK	A secure key chosen by RC for S_j
$h(.)$	A secure one-way hash function
$H(.)$	A bio-hash function
$\oplus$	An exclusive-OR operation
$\|\|$	The concatenation operation

doi:10.1371/journal.pone.0154308.t001

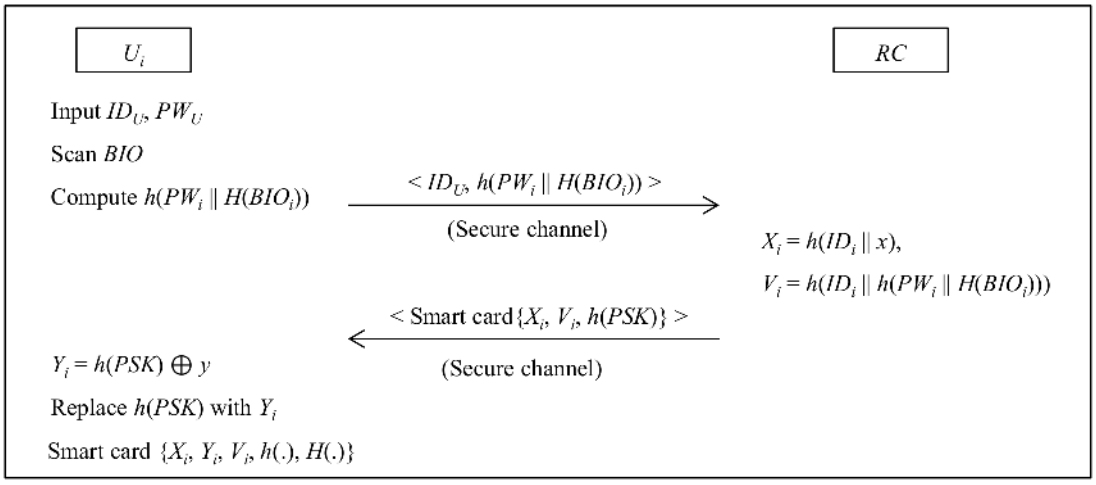

Fig 1. Registration phase of Lu et al.'s protocol.

doi:10.1371/journal.pone.0154308.g001

Registration phase

User (U_i) can register at registration center (RC) for the first time as shown in Fig 1.

Step 1: U_i chooses an identity ID_i, password PW_i and computes $h(PW_i \| H(BIO_i))$. Then sends a request message $< ID_i, h(PW_i \| H(BIO_i)) >$ to RC via a secure channel.

Step 2: RC computes $X_i = h(ID_i \| x)$, $V_i = h(ID_i \| h(PW_i \| H(BIO_i)))$. Then RC stores the parameters $\{X_i, V_i, h(PSK)\}$ on a smartcard and delivers it to U_i via a secure channel.

Step 3: Upon receiving the smartcard from RC, U_i computes $Y_i = h(PSK) \oplus y$, and replaces $h(PSK)$ with Y_i. Thus, the smartcard contains $\{X_i, Y_i, V_i, h(.)\}$.

Login and authentication phases

In this phase, user (U_i) and server (S_j) authenticates each other, and also establishes a session between them as shown in Fig 2. U_i can launch the login request by inserting smartcard, inputs ID_i, PW_i and BIO_i.

Step 1: Smartcard computes $h(PW_i \| H(BIO_i))$ and then verifies the condition $V_i \stackrel{?}{=} h(ID_i \| h(PW_i \| H(BIO_i)))$. If it generates negative result, the login request can be terminated.

Step 2: Smartcard generates a random number n_1, timestamp T_1 and computes $K = h((Y_i \oplus y) \| SID_j)$, $M_1 = K \oplus ID_i$, $M_2 = n_1 \oplus K$, $M_3 = K \oplus h(PW_i \| H(BIO_i))$, $Z_i = h(X_i \| n_1 \| h(PW_i \| H(BIO_i)) \| T_1)$, and sends the request message $< Z_i, M_1, M_2, M_3, T_1 >$ to S_j.

Step 3: S_j checks the freshness of the request message by verifying $T_c - T_1 \leq \Delta T$. If it holds, then S_j computes $K = h(h(PSK) \| SID_j))$ to retrieve $ID_i = K \oplus M_1$, $n_1 = M_2 \oplus K$, $h(PW_i \| H(BIO_i)) = K \oplus M_3$. Now, S_j computes $X_i = h(ID_i \| x)$ and verifies $Z_i \stackrel{?}{=} h(X_i \| n_1 \| h(PW_i \| H(BIO_i)) \| T_1)$. If the condition holds, then S_j authenticates U_i, otherwise process aborts.

Step 4: S_j further generates a random number n_2, timestamp T_2 and computes $SK_{ji} = h(n_1 \| n_2 \| K \| X_i)$, $M_4 = n_2 \oplus h(n_1 \| h(PW_i \| H(BIO_i)) \| X_i)$, $M_5 = h(ID_i \| n_1 \| n_2 \| K \| T_2)$. S_j sends the response $< M_4, M_5, T_2 >$ to U_i.

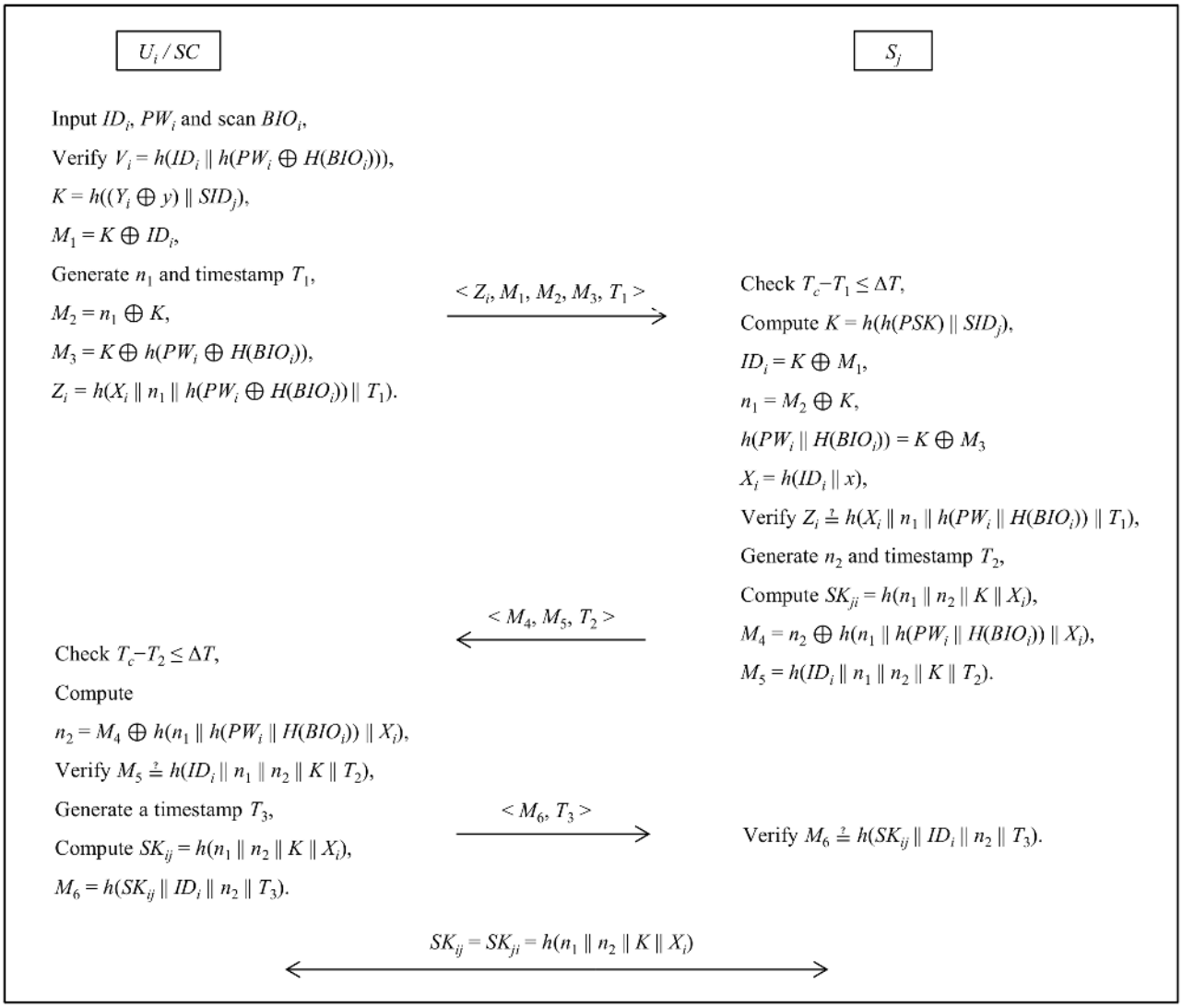

Fig 2. Login and authentication phases of Lu et al.'s protocol.

doi:10.1371/journal.pone.0154308.g002

Step 5: U_i checks the freshness of the message by verifying $T_c - T_2 \leq \Delta T$. If it holds, then U_i computes $n_2 = M_4 \oplus h(n_1 \| h(PW_i \| H(BIO_i)) \| X_i)$ and checks $M_5 \stackrel{?}{=} h(ID_i \| n_1 \| n_2 \| K \| T_2)$. If it generates positive result, then U_i authenticates S_j, otherwise process aborts.

Step 6: U_i generates a timestamp T_3 and computes $SK_{ij} = h(n_1 \| n_2 \| K \| X_i)$, $M_6 = h(SK_{ij} \| ID_i \| n_2 \| T_3)$. Finally, U_i sends $< M_6, T_3 >$ to S_j.

Step 7: S_j verifies the freshness of T_3 and $M_6 \stackrel{?}{=} h(SK_{ij} \| ID_i \| n_2 \| T_3)$. If it holds, then the mutual authentication with key agreement process between U_i and S_j is completed.

Password changing phase

A user (U_i) can update his/her existing password with new one without the help of registration center (RC) as explained below.

Step 1: U_i inserts smartcard, inputs the identity ID_i, password PW_i, scans the biometrics BIO_i and then verifies the condition $V_i \stackrel{?}{=} h(ID_i \,||\, h(PW_i \,||\, H(BIO_i)))$. If it holds, then U_i is allowed to choose a new password PW_i^{new}.

Step 2: Smartcard computes $V_i^{new} \stackrel{?}{=} h(ID_i \,||\, h(PW_i^{new} \,||\, H(BIO_i)))$ and replaces existing V_i with V_i^{new}.

Cryptanalysis of Lu et al.'s Protocol

This section cryptanalyses Lu et al.'s [40] protocol and provides a detailed discussion of all security limitations. Lu et al. asserted that their protocol can withstand several renowned attacks while achieving important security features. Conversely, this section proves that their protocol consists of significant drawbacks.

Limitation 1: Prone to server impersonation attack

In Lu et al.'s protocol, an adversary *Æ* can impersonate as a legitimate server as elucidated here. During server registration phase of Lu et al.'s protocol, registration center *RC* shares the chosen secret key *PSK* and random number x with S_j via a secure channel. When an adversary's server *Æ* registers with the *RC*, then after he/she can act as a legitimate server and access all the user's valuable data due to the possession of common shared attributes x and *PSK* in following way:

Step 1: During login and authentication phase, U_i launches the authentication request $< Z_i, M_1, M_2, M_3, T_1 >$ by inserting smartcard and inputting ID_i, PW_i and BIO_i.

Step 2: Upon receiving the request from U_i, *Æ* computes $K = h(h(PSK) \,||\, SID_j))$, $ID_i = K \oplus M_1$, $n_1 = M_2 \oplus K$, $h(PW_i \,||\, H(BIO_i)) = K \oplus M_3$, $X_i = h(ID_i \,||\, x)$.

Step 3: Now, *Æ* generates a random number n_2, timestamp T_2 and computes $SK_{ji} = h(n_1 \,||\, n_2 \,||\, K \,||\, X_i)$, $M_4 = n_2 \oplus h(n_1 \,||\, h(PW_i \,||\, H(BIO_i)) \,||\, X_i)$, $M_5 = h(ID_i \,||\, n_1 \,||\, n_2 \,||\, K \,||\, T_2)$. *Æ* sends the response $< M_4, M_5, T_2 >$ to U_i.

Step 4: U_i checks the freshness of the message by computing $T_c - T_2 \leq \Delta T$. If it holds, then U_i computes $n_2 = M_4 \oplus h(n_1 \,||\, h(PW_i \,||\, H(BIO_i)) \,||\, X_i)$ and verifies $M_5 \stackrel{?}{=} h(ID_i \,||\, n_1 \,||\, n_2 \,||\, K \,||\, T_2)$. It is obvious that the condition holds, consequently U_i treats *Æ* as legitimate S_j.

Step 5: U_i generates a timestamp T_3 and computes $SK_{ij} = h(n_1 \,||\, n_2 \,||\, K \,||\, X_i)$, $M_6 = h(SK_{ij} \,||\, ID_i \,||\, n_2 \,||\, T_3)$. Finally, U_i sends $< M_6, T_3 >$ to *Æ*.

Now, U_i may start the communication with *Æ* using the computed session key $SK_{ij} = h(n_1 \,||\, n_2 \,||\, K \,||\, X_i)$ but then is unaware of impersonation attack by *Æ*.

Limitation 2: Prone to man-in-middle attack

Lu et al.'s protocol is susceptible to man-in-middle attack while disclosing user's personal valuable data such as ID_i and $h(PW_i \,||\, H(BIO_i))$ as presented here. Assume a legitimate user who contains $h(PSK)$ becomes an adversary *Æ*, then he/she can cause possible damage to the system which is explained in the prone to user impersonation attack subsection.

Step 1: Consider a scenario where *Æ* capture the U_i's message $< Z_i, M_1, M_2, M_3, T_1 >$ while sending to S_j during authentication phase.

Step 2: *Æ* can compute $K = h(h(PSK) \,||\, SID_j))$ and obtain $ID_i = K \oplus M_1$, $n_1 = M_2 \oplus K$, $h(PW_i \,||\, H(BIO_i)) = K \oplus M_3$ by using $h(PSK)$ and openly available SID_j values.

Step 3: Now *Æ* comprises U_i's personal identifiable information such as ID_i and $h(PW_i \,||\, H(BIO_i))$ which are very unique.

Limitation 3: Prone to user impersonation attack

In a remote user communication protocol, anyone shall be treated as a legitimate user of the network if he/she has valid authentication credentials or could be able to construct a valid authentication request message. In Lu et al.'s protocol, an adversary *Æ* can impersonate a valid user as explained below.

Step 1: As enlightened in prone to server impersonation attack and man-in-middle subsections, *Æ* can obtain U_i's personal identifiable information such as ID_i and $h(PW_i \,||\, H(BIO_i))$, and possesses x and PSK values.

Step 2: Now, *Æ* generates a random number n_1, timestamp T_1 and computes $K = h(h(PSK) \,||\, SID_j))$, $M_1 = K \oplus ID_i$, $M_2 = n_1 \oplus K$, $M_3 = K \oplus h(PW_i \,||\, H(BIO_i))$, $X_i = h(ID_i \,||\, x)$, $Z_i = h(X_i \,||\, n_1 \,||\, h(PW_i \,||\, H(BIO_i)) \,||\, T_1)$. *Æ* sends the request $< Z_i, M_1, M_2, M_3, T_1 >$ to S_j.

Step 3: S_j checks the freshness of the message by computing $T_c - T_1 \leq \Delta T$. If it holds, then S_j computes $K = h(h(PSK) \,||\, SID_j)$ to retrieve $ID_i = K \oplus M_1$, $n_1 = M_2 \oplus K$, $h(PW_i \,||\, H(BIO_i)) = K \oplus M_3$. Then, S_j computes $X_i = h(ID_i \,||\, x)$ and verifies $Z_i = h(X_i \,||\, n_1 \,||\, h(PW_i \,||\, H(BIO_i)) \,||\, T_1)$. It is obvious that all the conditions generates positive results and S_j treats *Æ* as legitimate U_i and proceeds further.

Step 4: S_j generates a random number n_2, timestamp T_2 and computes $SK_{ji} = h(n_1 \,||\, n_2 \,||\, K \,||\, X_i)$, $M_4 = n_2 \oplus h(n_1 \,||\, h(PW_i \,||\, H(BIO_i)) \,||\, X_i)$, $M_5 = h(ID_i \,||\, n_1 \,||\, n_2 \,||\, K \,||\, T_2)$. S_j sends the response $< M_4, M_5, T_2 >$ to *Æ*.

Step 5: *Æ* computes $n_2 = M_4 \oplus h(n_1 \,||\, h(PW_i \,||\, H(BIO_i)) \,||\, X_i)$ and verifies $M_5 \stackrel{?}{=} h(ID_i \,||\, n_1 \,||\, n_2 \,||\, K \,||\, T_2)$. Now, *Æ* generates a timestamp T_3 and computes $SK_{ij} = h(n_1 \,||\, n_2 \,||\, K \,||\, X_i)$, $M_6 = h(SK_{ij} \,||\, ID_i \,||\, n_2 \,||\, T_3)$. Finally, *Æ* sends $< M_6, T_3 >$ to S_j.

Step 6: S_j verifies the freshness of T_3 and $M_6 \stackrel{?}{=} h(SK_{ij} \,||\, ID_i \,||\, n_2 \,||\, T_3)$. Since M_6 holds, S_j completes mutual authentication and allows *Æ* to access network services.

Limitation 4: Lack of user anonymity

Lu et al. claimed that their protocol can achieve one of the important security features called user anonymity. On the contrary, this subsection shows that their protocol cannot hold user anonymity property unlike their claim. During login and authentication phase, U_i transmits the authentication request message $< Z_i, M_1, M_2, M_3, T_1 >$ to S_j over public channels. The transmitted parameter $M_1 = K \oplus ID_i$, where $K = h((Y_i \oplus y) \,||\, SID_j))$ in the message $< Z_i, M_1, M_2, M_3, T_1 >$ are unique for each user and static during all logins. Hence anyone can track the actions of valid users, if he/she captures M_1 value.

Limitation 5: Lack of perfect forward secrecy

Forward secrecy ensures that session key is remaining safe, even if the long term private keys of communicating parties are compromised. In Lu et al. protocol, session key is computed as $SK_{ij} = h(n_1 \,||\, n_2 \,||\, K \,||\, X_i)$. The involved parameters K and X_i are dependent on long term secret keys, and n_1 and n_2 are random numbers. As proved in server impersonation attack, when S_j's long term secret keys x and PSK are compromised, then *Æ* can compute $K = h(h(PSK) \,||\, SID_j))$, $X_i = h(ID_i \,||\, x)$ and derive session key $SK_{ij} = h(n_1 \,||\, n_2 \,||\, K \,||\, X_i)$ subsequently. Therefore, Lu et al. protocol does not achieve another vital security feature called perfect forward secrecy.

Limitation 6: Prone to clock synchronization problem

Lu et al. uses timestamps to avoid replay attacks on their protocol. The messages $< Z_i, M_1, M_2, M_3, T_1 >$ and $< M_4, M_5, T_2 >$ transmitted between U_i and S_j contains timestamps T_1 and T_2. Upon receiving these messages S_j and U_i checks the validity of timestamps by computing $T_c - T_1 \leq \Delta T$ and $T_c - T_2 \leq \Delta T$. If these holds, then only S_j and U_i proceeds further to

authenticate each other. In the current world, millions of users contain computing devices due to the greater deployment of networks and technology. It is extremely difficult to synchronize the local system clocks of such a large set of communicating devices. Even a tiny difference in the time could lead to failure of authenticating users. Thus, Lu et al.'s protocol is prone to clock synchronization problem.

The Proposed Protocol

This section proposes a lightweight biometric based remote mutual authentication with key agreement protocol for multi-server architecture using elliptic curve cryptography. The proposed protocol comprises three participants: user (U_i), application server (AS), registration server (RS) and six phases: registration server initialization phase, application server registration phase, user registration phase, login phase, mutual authentication with key agreement phase, and password and biometrics changing phase. The notations used in the proposed protocol are listed in Table 2.

Registration server initialization phase

Registration server (RS) generates following parameters in order to initialize the system.

Step 1: RS chooses an elliptic curve equation E with an order n.

Step 2: RS selects a base point P over E and chooses a one-way cryptographic hash function $h(.)$.

Step 3: RS publishes the information $\{E, P, h(.)\}$.

Application server registration phase

In this phase, application server (AS) sends a registration request to the registration server (RS) in order to become an authorized server. The application server registration process consists of following steps:

Step 1: AS computes public key $R_S = x_S P$ sends registration request $< SID_S, R_S >$ to the RS.

Step 2: RS computes $K_S = h(SID_S \,||\, PSK)$, where PSK is RS's secret key for application servers and RS stores $\{SID_S, R_S, K_S\}$ in its database table T_{US}.

Step 3: RS sends K_S to AS, which can be used in further phases of authentication.

Table 2. Notations of the proposed protocol.

U_i	An i^{th} user
AS	Application server
RS	Registration server
ID_U	Identity of U_i
PW_U	Password of U_i
b	A number chosen by U_i for registration
ID_S	Identity of AS
USK	A secure key chosen by RS for U_i
PSK	A secure key chosen by RS for AS
x_U, x_S, N_1	Random numbers chosen by U_i and AS
$h(.)$	A secure one-way hash function
$H(.)$	A bio-hash function
$\oplus$	An exclusive-OR operation
$\|\|$	The concatenation operation

doi:10.1371/journal.pone.0154308.t002

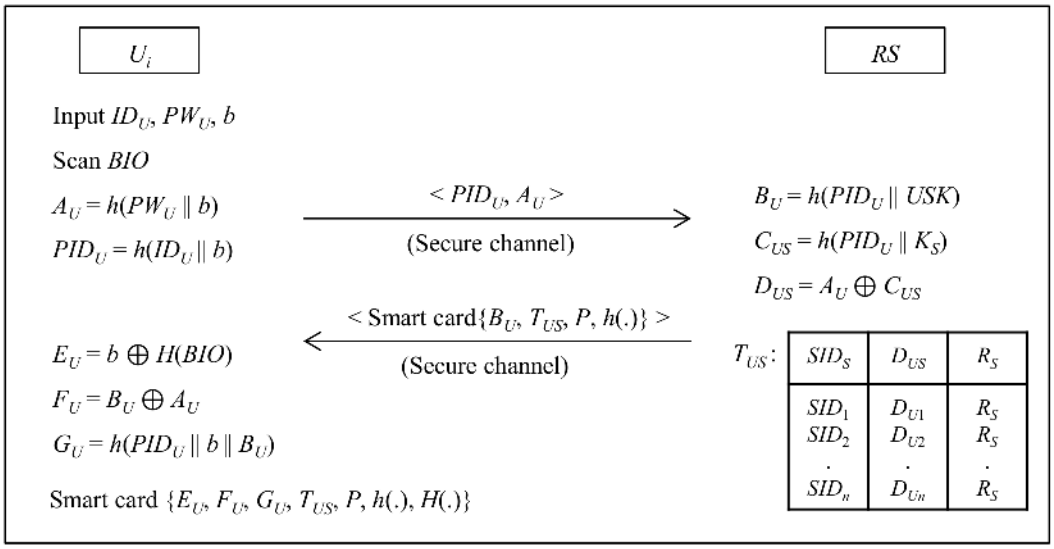

Fig 3. User registration phase.

doi:10.1371/journal.pone.0154308.g003

User registration phase

A new user (U_i), who wants to avail the services provided by application servers must register with registration server (RS). U_i goes after the following steps to register at RS as shown in Fig 3.

Step 1: U_i chooses an identity ID_U, password PW_U, a number b and scans biometrics BIO and computes $A_U = h(PW_U \| b)$, $PID_U = h(ID_U \| b)$. U_i sends a request message $< PID_U, A_U >$ to RS via a secure channel.

Step 2: RS computes $B_U = h(PID_U \| USK)$, $C_{US} = h(PID_U \| K_S)$ and $D_{US} = A_U \oplus C_{US}$, where USK is RS's secret key for users.

Step 3: RS personalize the parameters $\{B_U, T_{US}, P, h(.)\}$ on a smartcard and delivers it to U_i via a secure channel.

Step 4: U_i computes $E_U = b \oplus H(BIO)$, $F_U = B_U \oplus A_U$, $G_U = h(PID_U \| b \| B_U)$ and stores E_U, G_U, F_U on the received smart card after deleting B_U from smartcard. Thus the smartcard finally contains the parameters $\{E_U, G_U, F_U, T_{US}, P, h(.), H(.)\}$.

Login phase

When a user (U_i) wants to access the services of application server (AS), he/she launches the login request by inserting smartcard (SC), and inputting ID_U, PW_U and BIO.

Step 1: $SC \rightarrow AS$: $< AID_U, M_1, R_U >$

SC computes $b = E_U \oplus H(BIO)$, $A_U = h(PW_U \| b)$, $PID_U = h(ID_U \| b)$, $B_U = F_U \oplus A_U$ and then verifies the condition $G_U \stackrel{?}{=} h(PID_U \| b \| B_U)$. If it generates negative result, the login request can be terminated. Otherwise, SC retrieves corresponding application server's D_{US} and R_S values from the table T_{US}. Then, SC generates a random number x_U and calculates $R_U = x_U P$, $R_U' = x_U R_S$, $C_{US} = A_U \oplus D_{US}$, $AID_U = PID_U \oplus R_U'$, $M_1 = h(PID_U \| C_{US} \| R_U \| R_U')$ and sends the login request message $< AID_U, M_1, R_U >$ to AS.

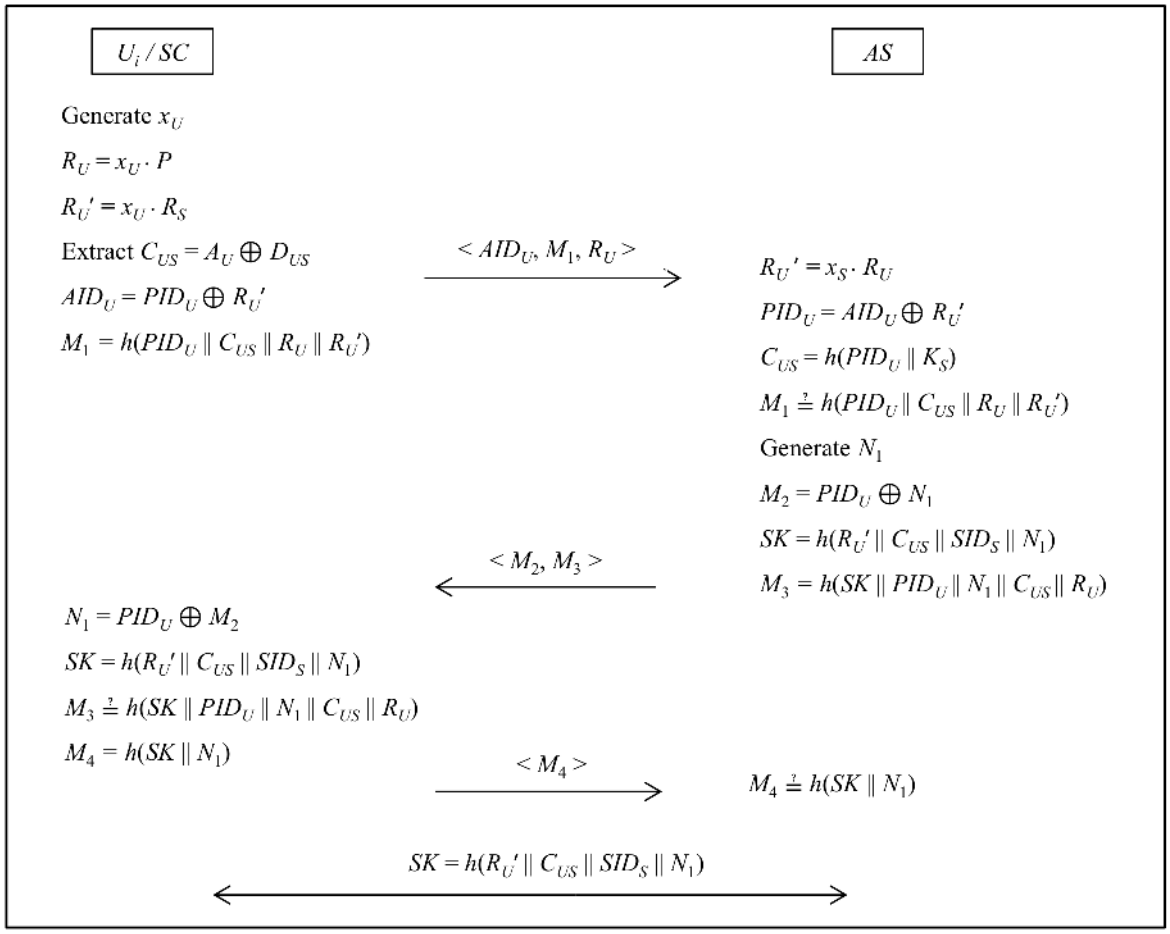

Fig 4. Mutual authentication with key-agreement phase.

doi:10.1371/journal.pone.0154308.g004

Mutual authentication with key-agreement phase

In this phase, U_i and AS authenticates each other and computes a session key for further secure communication over public channels. The entire mutual authentication with key agreement phase is illustrated in Fig 4.

Step 1: AS computes $R_U' = x_S R_U$, $PID_U = AID_U \oplus R_U'$, $C_{US} = h(PID_U \| K_S)$ and verifies the condition $M_1 \stackrel{?}{=} h(PID_U \| C_{US} \| R_U \| R_U')$. If the condition holds, AS authenticates U_i, otherwise the process can be terminated.

Step 2: $AS \rightarrow SC$: $< M_2, M_3 >$
AS further generates a random number N_1 and computes $SK = h(R_U' \| C_{US} \| SID_S \| N_1)$, $M_2 = PID_U \oplus N_1$, $M_3 = h(SK \| PID_U \| N_1 \| C_{US} \| R_U)$. AS sends $< M_2, M_3 >$ to SC.

Step 3: $SC \rightarrow AS$: $< M_4 >$
SC computes $N_1 = PID_U \oplus M_2$, $SK = h(R_U' \| C_{US} \| SID_S \| N_1)$ and verifies the condition $M_3 \stackrel{?}{=} h(SK \| PID_U \| N_1 \| C_{US} \| R_U)$. If the condition holds, U_i authenticates AS, otherwise the process can be terminated. Then, SC computes $M_4 = h(SK \| N_1)$ and sends it to AS.

Step 4: AS verifies $M_4 \stackrel{?}{=} h(SK \| N_1)$ and reconfirms the authenticity of U_i. Now, U_i and AS can start communication with the computed session key SK.

Password and biometrics changing phase

This procedure invokes when a user (U_i) wish to update his/her existing password with new one. In this procedure, U_i can change his/her password without the involvement of registration server (RS) as follows:

Step 1: U_i inserts smartcard SC and inputs ID_U, PW_U and BIO.

Step 2: SC computes $b = E_U \oplus H(BIO)$, $A_U = h(PW_U \,||\, b)$, $PID_U = h(ID_U \,||\, b)$, $B_U = F_U \oplus A_U$ and then verifies the condition $G_U \stackrel{?}{=} h(PID_U \,||\, b \,||\, B_U)$. If the condition holds, U_i derives $C_{US} = A_U \oplus D_{US}$ for all the servers in the table T_{US}, otherwise request can be dropped.

Step 3: U_i chooses a new password $PW_U{}^{\#}$ and $BIO^{\#}$ and then computes $A_U{}^{\#} = h(PW_U{}^{\#} \,||\, b)$, $F_U{}^{\#} = B_U \oplus A_U{}^{\#}$, $D_{US}{}^{\#} = A_U{}^{\#} \oplus C_{US}$ and $E_U{}^{\#} = b \oplus H(BIO^{\#})$. U_i updates the table $T_{US}{}^{\#}$ and the parameters $F_U{}^{\#}$, $E_U{}^{\#}$ on the smartcard. Thus the smartcard finally contains the parameters $\{E_U{}^{\#}, F_U{}^{\#}, G_U, T_{US}{}^{\#}, P, h(.), H(.)\}$.

Security Analysis

This section exhibits the security analysis of proposed authentication protocol for multi-server architecture by describing each security feature. This analysis checks various security aspects and ensures that the proposed protocol is resistant to different attacks and certain flaws are not exhibited.

Formal security analysis using BAN logic

Formal security analysis of the proposed protocol is verified with the help of Burrows-Abadi-Needham (BAN) logic [46]. This section proves that the proposed protocol provides secure mutual authentication between a user U_i and an application server AS.

The following notations are used in formal security analysis using the BAN logic:

- $Q \mid\equiv X$: Principal Q believes the statement X.
- $\#(X)$: Formula X is fresh.
- $Q \mid \Rightarrow X$: Principal Q has jurisdiction over the statement X.
- $Q \lhd X$: Principal Q sees the statement X.
- $Q \mid \sim X$: Principal Q once said the statement X.
- (X, Y): Formula X or Y is one part of the formula (X, Y).
- $\langle X \rangle_Y$: Formula X combined with the formula Y.
- $Q \overset{K}{\leftrightarrow} R$: Principal Q and R may use the *shared key* K to communicate among each other. The key K is good, in that it will never be discovered by any principal except Q and R.
- $Q \overset{X}{\Leftrightarrow} R$: Formula X is secret known only to Q and R, and possibly to principals trusted by them.

In addition, the following four BAN logic rules are used to prove that the proposed protocol provides a secure mutual authentication between U_i and AS:

- Rule 1. Message-meaning rule: $\frac{R \mid\equiv R \overset{Y}{\leftrightarrow} S, \quad R \lhd \langle X \rangle_Y}{R \mid\equiv S \mid\sim X}$ and $\frac{P \mid\equiv P \overset{Y}{\leftrightarrow} Q, \quad P \lhd \langle X \rangle_Y}{P \mid\equiv Q \mid\sim X}$
- Rule 2. Nonce-verification rule: $\frac{R \mid\equiv \#(X), \quad R \mid\equiv S \mid\sim X}{R \mid\equiv S \mid\equiv X}$

- Rule 3. Jurisdiction rule: $\frac{R \mid\equiv S \mid\Rightarrow X,\quad R \mid\equiv S \mid\equiv X}{R \mid\equiv X}$
- Rule 4. Freshness-conjuncatenation rule: $\frac{R \mid\equiv \#(X)}{R \mid\equiv \#(X,Y)}$

In order to show that the proposed protocol provides secure mutual authentication between a node R in the cluster C_i and TM, we need to achieve the following four test goals:

- Goal 1: $U_i \mid \equiv U_i \overset{SK}{\leftrightarrow} AS$
- Goal 2: $U_i \mid \equiv AS \mid \equiv U_i \overset{SK}{\leftrightarrow} AS$
- Goal 3: $AS \mid \equiv U_i \overset{SK}{\leftrightarrow} AS$
- Goal 4: $AS \mid \equiv U_i \mid \equiv U_i \overset{SK}{\leftrightarrow} AS$

Generic form: The generic forms of the transmitted messages between the user U_i and the application server AS in the proposed protocol are given below:

- M1. $U_i \rightarrow AS$: $\langle PID_U, R_U, R_U' \rangle_{C_{US}}$
- M2. $AS \rightarrow U_i$: $\langle SK, PID_U, N_1, R_U \rangle_{C_{US}}$
- M3. $U_i \rightarrow AS$: $\langle R_U', SID_S, N_1 \rangle_{C_{US}}$

Note that the message M3, $U_i \rightarrow AS$: $h(h(R_U', C_{US}, SID_S, N_1), N_1)$ authenticates the parameters R_U', SID_S, N_1 under the shared secret C_{US} between U_i and AS. Thus, for simplicity we assume that the message M3 as $U_i \rightarrow AS$: $\langle R_U', SID_S, N_1 \rangle_{C_{US}}$ in the generic form.

Idealized form: The arrangement of the transmitted messages between U_i and AS in the proposed protocol to the idealized forms are as follows:

- M1. $U_i \rightarrow AS$: $\langle PID_U, R_U, U_i \overset{R_U'}{\leftrightarrow} AS \rangle_{U_i \overset{C_{US}}{\leftrightarrow} AS}$
- M2. $AS \rightarrow U_i$: $\langle U_i \overset{SK}{\leftrightarrow} AS, PID_U, N_1, R_U \rangle_{U_i \overset{C_{US}}{\leftrightarrow} AS}$
- M3. $U_i \rightarrow AS$: $\langle U_i \overset{R_U'}{\leftrightarrow} AS, SID_S, N_1 \rangle_{U_i \overset{C_{US}}{\leftrightarrow} AS}$

Hypotheses: The following are the initial assumptions of the proposed protocol:

- H1: $U_i \mid\equiv \#(R_U)$,
- H2: $AS \mid\equiv \#(N_1)$,
- H3: $U_i \mid \equiv U_i \overset{C_{US}}{\leftrightarrow} AS$,
- H4: $AS \mid \equiv U_i \overset{C_{US}}{\leftrightarrow} AS$,
- H5: $AS \mid \equiv U_i \mid \Rightarrow U_i \overset{R_U'}{\leftrightarrow} AS$,
- H6: $U_i \mid \equiv AS \mid \Rightarrow U_i \overset{SK}{\leftrightarrow} AS$,
- H7: $U_i \mid \equiv U_i \overset{R_U'}{\leftrightarrow} AS$.

In the following, we prove the above test goals in order to show the secure authentication using the BAN logic rules and the assumptions.

- From the message M1, we have, S1: $AS \triangleleft \langle PID_U, R_U, U_i \overset{R_U'}{\leftrightarrow} AS \rangle_{U_i \overset{C_{US}}{\leftrightarrow} AS}$

- From S1, H4, and Rule 1, we get, S2: $AS| \equiv U_i| \sim \langle PID_U, R_U, U_i \overset{R_U'}{\leftrightarrow} AS \rangle$
- From the message M2, we have, S3: $U_i \lhd U_i\langle \overset{SK}{\leftrightarrow} AS, PID_U, N_1, R_U\rangle_{U_i \overset{C_{US}}{\Longleftrightarrow} AS}$
- From S3, H3, and Rule 1, we obtain, S4: $U_i| \equiv AS| \sim \langle U_i \overset{SK}{\leftrightarrow} AS, PID_U, N_1, R_U\rangle$
- From S4, H1, Rule 2, and Rule 4, we get, S5: $U_i| \equiv AS| \equiv U_i \overset{SK}{\leftrightarrow} AS$ (Goal 2)
- From S5 and Jurisdiction rule Rule 3, we obtain, S6: $U_i| \equiv U_i \overset{SK}{\leftrightarrow} AS$ (Goal 1)
- From the message M3, we have, S7: $AS \lhd \langle U_i \overset{R_U'}{\leftrightarrow} AS, SID_S, N_1\rangle_{U_i \overset{C_{US}}{\Longleftrightarrow} AS}$
- From the message S7, H4, and Rule 1, we have, S8: $AS| \equiv U_i| \sim \langle U_i \overset{R_U'}{\leftrightarrow} AS, SID_S, N_1\rangle$
- From S8, H2, Rule 2, and Rule 4, we get, S9: $AS| \equiv U_i| \equiv U_i \overset{R_U'}{\leftrightarrow} AS$
- Since SK is computed as $SK = h(h(R_U', C_{US}, SID_S, N_1), N_1)$, from S9 and H4, we get the required goal Goal 3, as S10: $AS| \equiv U_i| \equiv U_i \overset{SK}{\leftrightarrow} AS$ (Goal 4)
- Finally, from S10 and Jurisdiction rule Rule 3, we obtain, S11: $AS| \equiv U_i \overset{SK}{\leftrightarrow} AS$ (Goal 3)

Informal security analysis

Proposition 1. The proposed protocol achieves user anonymity and untraceability.

Proof. The proposed protocol does not reveal the real identities of users throughout all the phases of communication. In the user registration phase U_i submits pseudonym identity $PID_U = h(ID_U || b)$ and the real identity is guarded with a one-way hash function. During login phase, the pseudonym identity PID_U is converted as anonymous in the form of $AID_U = PID_U \oplus R_U'$. The identity is dynamic for every login, due to its association with a randomly chosen number x_U, where $R_U = x_U P$ and $R_U' = x_U R_S$. An adversary cannot retrieve the user's pseudonym identity PID_U without having the knowledge of the user's password PW_U and b. Moreover, it is believed to be impossible to compute R_U' from R_U and R_S due to the fact of ECDLP. The proposed protocol provides another important feature called untraceability. An adversary may try to trace the actions of users by observing the transmitting parameters. In the login phase, U_i sends the message $< AID_U, M_1, R_U >$ to AS. All the parameters are dynamic and does not disclose any information about U_i. Thus, the proposed protocol achieves user anonymity with untraceability.

Proposition 2. The proposed protocol is secure against replay attacks and clock synchronization problem.

Proof. The proposed protocol adopts the method discussed in the recent protocols [2] [3] [24] [25] [36], known as deployment of random numbers to endure replay attack. During mutual authentication and key-agreement phase, AS and U obtains $\{PID_U, R_U\}$ and $\{N_1\}$ and stores the values in its database tables, respectively. Consider a scenario where an adversary acquire $\{AID_U, M_1, R_U\}$ or $\{M_2, M_3\}$ or $\{M_4\}$ and replay the same message. However, adversary definitely cannot construct a valid session due to the reason explained here. All the captured parameters are randomized by incorporating a random number x_U in the form of $R_U = x_U P$ and $R_U' = x_U R_S$ in $AID_U = PID_U \oplus R_U'$, $M_1 = h(PID_U || C_{US} || R_U || R_U')$. The random number x_U always keeps the transmitting parameters as dynamic for every session. If an adversary sends $< AID_U, M_1, R_U >$ to AS, it identifies R_U as previous transmitted message and drops the requested session. In the same way, N_1 helps in identifying the replay attacks of $\{M_2, M_3\}$ and $\{M_4\}$.

In a cryptographic authentication protocol environment, timestamps are used to protect the messages from replay attacks. Basically, timestamps will be generated from the internal clocks

of computing systems and may differ from system to system known as, clock synchronization problem. Hence, in the current large network field, time stamps are not the definite solutions due to clock synchronization problems. As an alternative possible solution, the proposed protocol deploys random numbers x_U and N_1.

Proposition 3. The proposed protocol is secure against stolen smart card attack.

Proof. Reading a smartcard stored values is possible by means of power analysis and various other ways [49] [50] [51]. Assume a valid user's smartcard is stolen by an adversary and stored parameters $\{E_U, G_U, F_U, T_{US}, P, h(.)\}$ on it are extracted. Now, the adversary may try to derive authentication credentials from the extracted parameters. However, adversary undeniably cannot obtain any valuable information from these values, since all the important parameters such as $E_U = b \oplus H(BIO)$, $F_U = B_U \oplus A_U$, $G_U = h(PID_U \,||\, b \,||\, B_U)$ are safeguarded with a one-way hash function, where $PID_U = h(ID_U \,||\, b)$, $A_U = h(PW_U \,||\, b)$ and $B_U = h(PID_U \,||\, USK)$. Note that the identity of user ID_U is not stored on the smartcard. The adversary cannot obtain any login information using the smartcard stored parameters E_U, G_U, F_U. At the same time guessing the real identity ID_U and password PW_U is impractical. Aforementioned constraints proves that the proposed protocol is secure from smartcard stolen attack.

Proposition 4. The proposed protocol is secure against user impersonation attack.

Proof. Assume a situation where an adversary possesses a valid smartcard and wants to gain network access by perpetrating user impersonation attack. If an adversary wants to impersonate a legitimate user U_i, he/she requires to build a login request message $< AID_U, M_1, R_U >$, where $R_U = x_U P$, $AID_U = PID_U \oplus R_U'$, $M_1 = h(PID_U \,||\, C_{US} \,||\, R_U \,||\, R_U')$. Conversely, the adversary can barely compute two parameters $R_U^{\#} = x_U^{\#} P$ and $R_U^{\#\prime} = x_U^{\#} R_S$ by choosing his/her own random number $x_U^{\#}$. In order to compute rest of the two parameters, adversary requires user's identity ID_U and password PW_U, which are unobtainable.

On the other hand, the adversary should undergo login phase before making authentication request. During login phase, SC computes $b = E_U \oplus H(BIO)$, $A_U = h(PW_U \,||\, b)$, $PID_U = h(ID_U \,||\, b)$, $B_U = F_U \oplus A_U$ and then verifies the condition $G_U \stackrel{?}{=} h(PID_U \,||\, b \,||\, B_U)$. Unless the adversary enters the correct credentials, he/she cannot be allowed to further phases. Therefore, the adversary certainly requires legitimate identity ID_U and password PW_U for any furthermore computations. However, the probability of yielding correct ID_U and PW_U is negligible. The adversary may also try to extract PID_U from $AID_U = PID_U \oplus R_U'$, by guessing x_U from $R_U = x_U P$ and $R_U' = x_U R_S$. It is even more difficult to perform the above operation due to the fact of Elliptic Curve Diffie-Hellman Problem (ECDHP).

Proposition 5. The proposed protocol is secure against application server impersonation attack.

Proof. Usually, during authentication phase, AS computes $M_2 = PID_U \oplus N_1$, $SK = h(R_U' \,||\, C_{US} \,||\, SID_S \,||\, N_1)$, $M_3 = h(SK \,||\, PID_U \,||\, N_1 \,||\, C_{US} \,||\, R_U)$ with the generated random number N_1 and sends $< M_2, M_3 >$ to U_i. Consider a scenario where an adversary's server acts as a legitimate one and proceeds with the authentication and key agreement procedures. In order to compute session key SK, adversary must have R_U', C_{US} and SID_J. Assume that adversary still proceeds with the computations $R_U^{\#\prime} = x_S^{\#} R_U$, $M_2^{\#} = PID_U^{\#} \oplus N_1^{\#}$, $SK^{\#} = h(R_U^{\#\prime} \,||\, C_{US}^{\#} \,||\, SID_S \,||\, N_1^{\#})$, $M_3^{\#} = h(SK^{\#} \,||\, PID_U^{\#} \,||\, N_1^{\#} \,||\, C_{US}^{\#} \,||\, R_U)$ with the generated random numbers $N_1^{\#}$ and $x_S^{\#}$. Note that SID_S can be obtained from the table T_{US} in the smartcard. Upon receiving the response $< M_2^{\#}, M_3^{\#} >$, U_i computes $N_1^{\#} = PID_U \oplus M_2^{\#}$, $SK = h(R_U' \,||\, C_{US} \,||\, SID_S \,||\, N_1^{\#})$ and $M_3^{\#} = h(SK \,||\, PID_U \,||\, N_1^{\#} \,||\, C_{US} \,||\, R_U)$ and tallies the received $M_3^{\#}$ with the computed M_3. Here, U_i identifies it as a fake response from the malicious server due to $M_3 \neq M_3^{\#}$ and terminates the session immediately. Thus, the proposed protocol can withstand application server impersonation attacks.

Proposition 6. The proposed protocol is secure against man-in-middle attack.

Proof. In the proposed protocol scenario, adversary has the possibility of attacking either request message or response messages as elucidated here. Authentication request message $< AID_U, M_1, R_U >$ initiates from *SC* to *AS*. As explained in proposition 4, adversary can modify only one parameter $R_U^{\#} = x_U^{\#} P$ with the chosen random number $x_U^{\#}$. For instance the adversary sends the modified parameter in the message as $< AID_U, M_1, R_U^{\#} >$. Upon receiving it, *AS* computes $R_U^{\#\prime} = x_S R_U^{\#}$, $PID_U^{\#} = AID_U \oplus R_U^{\#\prime}$, $C_{US}^{\#} = h(PID_U^{\#} \,||\, K_S)$ and $M_1^{\#} = h(PID_U^{\#} \,||\, C_{US}^{\#} \,||\, R_U^{\#} \,||\, R_U^{\#\prime})$. Finally, *AS* compares the received M_1 with the computed $M_1^{\#}$ then apparently $M_1 \neq M_1^{\#}$. Accordingly, *AS* identifies it as a malicious attack and acknowledges the user U_i. In case, the adversary wants to accomplish active attacks on response messages either $< M_2, M_3 >$ or $< M_4 >$, then session key $SK = h(R_U' \,||\, C_{US} \,||\, SID_S \,||\, N_1)$ value and other parameters are essential and unobtainable. Thus the proposed protocol can withstand a man-in-middle attack.

Proposition 7. The proposed protocol is secure against password guessing attack.

Proof a. [*Offline password guessing attack*]: An adversary may attempt to guess the password PW_U from the extracted smart card stored parameters $\{E_U, G_U, F_U, T_{US}, P, h(.)\}$. The stored parameter $F_U = B_U \oplus A_U$ contains the password PW_U in the form $A_U = h(PW_U \,||\, b)$. An adversary can try to check the condition $F_U \stackrel{?}{=} B_U \oplus A_U$ while constantly guessing PW_U. In order to execute this, adversary needs ID_U and b values as well. However, ID_U value is nowhere stored and b value is protected with biometrics $H(BIO)$, which can neither be forged nor copied. The adversary may even attempt to perform the same on $AID_U = PID_U \oplus R_U'$ value intercepted from previous login message $< AID_U, M_1, R_U >$. To perform this, the adversary requires $R_U' = x_U R_S$. As a result, the adversary would fail to guess the correct password PW_U. Therefore, the proposed protocol is secure against offline password guessing attack.

Proof b. [*Online password guessing attack*]: If an adversary possesses the valid smartcard, he/ she may keep trying to login while guessing the password PW_U. Unless the adversary passes valid biometrics *BIO*; b value cannot be retrieved from E_U. Additionally, the login verification condition $G_U \stackrel{?}{=} h(PID_U \,||\, b \,||\, B_U)$ checks the correctness of all input credentials. If the adversary enters the wrong password for certain number of times, the system may abort and would not allow entering credentials for some time. In addition, it is almost impractical to guess all the required values within polynomial time.

Proposition 8. The proposed protocol is secure against privileged insider attack and does not maintain user verification table.

Proof. A privileged insider of the system can obtain the stored credentials of registered user and perpetrate malicious attacks subsequently. However, during user registration phase of proposed protocol, U_i does not submit identity ID_U and password PW_U in plaintext form to the registration server *RS*. U_i submits only $A_U = h(PW_U \,||\, b)$ and $PID_U = h(ID_U \,||\, b)$ to *RS* instead of original credentials, where b is a randomly chosen number. Hence, an insider cannot obtain the original credentials of any user. In this way, the proposed protocol attains resistance to insider attacks.

In the proposed protocol, *AS* authenticates U_i by verifying the equivalence of the received message with the computed values i.e. $M_1 \stackrel{?}{=} h(PID_U \,||\, C_{US} \,||\, R_U \,||\, R_U')$. Registration server *RS* is not involved in the authentication process, whereas password changing phase requires *RS*'s help. However, *RS* does not verify the legitimacy of U_i during this phase. Hence, *RS* does not require maintaining a database to store any kind of user's credentials. An intruder cannot be able to determine any information about users, while the servers are not maintaining user verification tables. Thus, the servers are free from investing on their storage spaces.

Proposition 9. The proposed protocol is secure against denial-of-service attack.

An adversary may cause denial-of-service attack, when he/she intercepts a valid authentication request message and replays the same message to *AS*. We have taken following approaches to prove that the proposed protocol is secure against denial-of-service attack.

Proof a. Consider a scenario where an adversary replays previous captured authentication request $\{AID_U, M_1, R_U\}$ without any modifications. Upon receiving the request, AS computes $R_U' = x_S \cdot R_U$, $PID_U = AID_U \oplus R_U'$, and compares the extracted $\{PID_U, R_U\}$ with the stored $\{PID_U, R_U\}$. When AS identify the received R_U is same as the stored R_U (i.e. $R_U == R_U$), then it can reject the request without even verifying $M_1 \stackrel{?}{=} h(PID_U \,||\, C_{US} \,||\, R_U \,||\, R_U')$. This procedure can be completed with just one elliptic curve point multiplication operation and one X-OR operation.

Proof b. Assume that an adversary transmits fake requests such as $\{AID_U^{\#}, M_1^{\#}, R_U^{\#}\}$ for multiple. Upon receiving this message, AS computes $R_U' = x_S \cdot R_U$, $PID_U = AID_U \oplus R_U'$, $C_{US} = h(PID_U \,||\, K_S)$, and verifies $M_1 \stackrel{?}{=} h(PID_U \,||\, C_{US} \,||\, R_U \,||\, R_U')$. It is obvious that the condition generates negative response due to unavailability of original C_{US} value with adversary. Therefore, AS believes it as a malicious attack and terminates the session, which requires two hash computations and one elliptic curve point multiplication operation.

Proposition 10. The proposed protocol provides forward secrecy.

Proof. Forward secrecy ensures that the session key remains safe, even though the long term private keys of communicating parties are compromised. The session key of the proposed protocol is computed as $SK = h(R_U' \,||\, C_{US} \,||\, SID_S \,||\, N_1)$ and the long term private key of the server K_S in $C_{US} = h(PID_U \,||\, K_S)$ is shielded with a hash function and is not possible to derive due to its one-way property. Although the long term key is compromised with an adversary; he/she still cannot construct a valid session key due to following reason. The parameter $R_U' = x_U \cdot R_S$ is dynamic due to its association with random generated number x_U, which is not possible to extract due to the reason of ECDLP. Therefore, the proposed protocol provides perfect forward secrecy.

Simulation for formal security verification using AVISPA tool

In this section, we simulate the proposed protocol using the widely accepted AVISPA for the formal security verification. For this purpose, we first provide a brief background of AVISPA tool and then the implementation details. We finally analyze the simulation results reported in this section. Note that AVISPA allows to verify whether a security protocol is safe or unsafe against replay and man-in-the-middle attacks. The main goal of the formal security verification simulation is to verify whether the proposed scheme is secure against replay and man-in-the-middle attacks.

Overview of AVISPA. AVISPA is a push-button tool for the automated validation of Internet security-sensitive protocols and applications. AVISPA is a widely-accepted and used tool to formally verify whether a cryptographic protocol is safe or unsafe against passive and active attacks including the replay and man-in-the-middle attacks [2], [52]. In AVIPSA, a security protocol is implemented using HLPSL (High Level Protocols Specification Language) [53], [54]. In HLPSL implementation, the basic roles are used for representing each participant role, and composition roles for representing scenarios of basic roles. The role system includes the number of sessions, the number of principals and the roles.

In HLPSL, an intruder (i) is modeled using the Dolev-Yao model [55] where the intruder can participate as a legitimate role. HLPSL is translated using HLPSL2IF translation to convert to the intermediate format (IF). IF is fed into one of the four backends: On-the-fly Model-Checker (OFMC), Constraint Logic based Attack Searcher (CL-AtSe), SAT-based Model-Checker (SATMC) and Tree Automata based on Automatic Approximations for the Analysis of Security Protocols (TA4SP). The detailed descriptions of these back-ends can be found in [54]. The output format (OF) is produced from IF by using one of these four back-ends. OF has the following sections [54]:

- SUMMARY: It indicates that whether the tested protocol is safe, unsafe, or inconclusive.
- DETAILS: It either explains under what condition the tested protocol is declared safe, or what conditions have been used for finding an attack, or finally why the analysis is inconclusive.
- PROTOCOL: It denotes the name of the protocol.
- GOAL: It indicates the goal of the analysis.
- BACKEND: It represents the name of the back-end used.
- At the end, after some comments and statistics, the trace of an attack (if any) is displayed in the standard Alice-Bob format.

There are several basic types supported in HLPSL, some of them are given below for better understanding of the implementation details in Section 7.2 [48]:

- *Agent*: It denotes the principal names. The intruder has always the special identifier *i*.
- *Public key*: It denotes agents' public keys in a public-key cryptosystem. For example, given a public (respectively private) key *pk*, its inverse private (respectively public) key *pr* is obtained by *inv*(*pk*).
- *Symmetric key*: It means the keys for a symmetric-key cryptosystem.
- *Text*: It is often used as nonces. These values can be also used for messages.
- *Nat*: It denotes the natural numbers in non-message contexts.
- *Const*: It denotes the constants.
- *Hash_func*: It represents cryptographic hash functions.

In HLPSL, for concatenation the associative "." operator is utilized. "*played_by X*" declaration means that the agent named in variable *X* plays in the role. A knowledge declaration (generally in the top-level *Environment* role) is used to specify the intruder's initial knowledge. Immediate reaction transitions are of the form $X = | > Y$, which relates an event *X* and an action *Y*. By the goal *secrecy_of P*, a variable *P* is kept permanently secret. Thus, if *P* is ever obtained or derived by the intruder, a security violation will result.

Various roles implementation in HLPSL. We have three basic roles: *user* for a user U_i, *registration server* for the registration server *RS* and *application server* for the application server *AS*. Besides these roles, the roles for the session, goal and environment in HLPSL are mandatory in the implementation. We have implemented the proposed protocol for user registration phase, login phase, and mutual authentication with key-agreement phase.

The role of the initiator, U_i is provided in Fig 5. U_i first receives the start signal, updates its state value from 0 to 1. The state value is maintained by the variable *State*. U_i sends the registration request message $< PID_U, A_U >$ securely to the *RS* during the user registration phase with the *SEND*() operation. U_i then receives a smart card *SC* containing the information $\{B_U, T_{US}, P, h()\}$ securely from the *RS* by the *RECV*() operation, and updates its state from 1 to 2.

In the login phase, U_i sends the message $< AID_U, M_1, R_U >$ to the *AS* via open channel. During the mutual authentication with key-agreement phase, U_i then receives the message $< M_2, M_3 >$ from the *AS* and sends reply $< M_4 >$ to the *AS* via open channel.

Note that *channel* (*dy*) declares that the channel is for the Dolev-Yao threat model [55]. The intruder (*i*) can thus intercept, analyze, and/or modify messages transmitted over the open channel. witness(A, B, id, E) declaration denotes for a (weak) authentication property of *A* by *B*

```
role user (Ui, RS, AS : agent,
        SKuirs : symmetric_key,
        % H is one−way hash function
        H: hash_func,
        SEND, RECV: channel(dy))
% Player: the user Ui
played_by Ui
def=
local State  : nat,
    B, Au, PIDu, IDu, PWu : text,
    USK, PSK, P, SIDs, Xu, Ru, Ruu : text,
    AIDu, M1, M4, Xs, N1 : text,
 F: hash_func
const ui_as_xu,
    s1, s2, s3: protocol_id
init  State := 0
transition
% User registration phase
1. State = 0 /\ RECV(start) =|>
State' := 1 /\ B' := new() /\ Au' := H(PWu.B')
        /\ PIDu' := H(IDu.B')
        /\ secret({IDu,B',PWu}, s1, {Ui})
% Send request message < PIDu, Au > to RS securely
        /\ SEND({PIDu'.Au'}_SKuirs)
% Receive smart card SC from RS securely
2. State = 1 /\ RECV({H(H(IDu.B').H(PWu.B').USK).SIDs.
            xor(H(PWu.B'),H(H(IDu.B').H(SIDs.PSK))).
            H(SIDs.PSK).P.H}_SKuirs) =|>
% Login phase
State' := 2 /\ secret({USK,PSK}, s2, RS)
        /\ Xu' := new() /\ Ru' := F(Xu'.P)
        /\ Ruu' := F(Xu'.F(Xs'.P))
        /\ AIDu' := xor(H(IDu.B'),Ruu')
        /\ M1' := H(H(IDu.B').H(H(IDu.B').H(SIDs.PSK)).
                Ru'.Ruu')
        /\ secret(Xs', s3, {RS, AS})
% Send message < AIDu, M1, Ru > to AS publicly
        /\ SEND(AIDu'.M1'.Ru')
% Ui has freshly generated random value x_U for AS
    /\ witness(Ui, AS, ui_as_xu, Xu')
% Receive message < M2, M3 > from AS publicly
3. State = 2 /\ RECV(xor(H(IDu.B'),N1').
            H(H(F(Xu'.F(Xs'.P)).H(H(IDu.B').H(SIDs.PSK)).
            SIDs.N1').H(IDu.B').N1'.H(H(IDu.B').H(SIDs.PSK)).
            F(Xu'.P))) =|>
% Send message < M4 > to AS publicly
State' := 3 /\ M4' :=
H(H(F(Xu'.F(Xs'.P)).H(H(IDu.B').H(SIDs.PSK)).
                SIDs.N1').N1')
        /\ SEND(M4')
% Ui's acceptance of the value N1 generated for Ui by AS
        /\ request(AS, Ui, as_ui_n1, N1')
end role
```

Fig 5. Role specification for user U_i.

doi:10.1371/journal.pone.0154308.g005

on *E*, declares that agent *A* is witness for the information *E*; this goal will be identified by the constant *id* in the goal section [48]. request(B, A, id, E) declaration represents a strong authentication property of *A* by *B* on *E*, declares that agent *B* requests a check of the value *E*; this goal will be identified by the constant id in the goal section [48]. For example, witness(U_i, AS, u_i as x_u, x_u') declares that U_i has freshly generated random number x_U for *AS*. By the declaration secret(ID_u, B', PW_u, s1, U_i), we mean that the information ID_U, *b* and PW_U are kept secret to U_i only, which is identified by the protocol id s1.

In a similar way, the roles of the *AS* and *RS* of the proposed protocol are implemented and shown in Figs 6 and 7, respectively. The declaration, request(U_i, AS, u_i as x_u, x_u'), signifies the *AS*'s acceptance of the value x_U generated for *AS* by U_i. The roles for the goal and environment, and the session of the proposed protocol are also shown in Figs 8 and 9, respectively. In the session role, all the basic roles including *user*, *registrationserver* and *applicationserver* are the instances with concrete arguments. The top-level role (environment) is always specified in the HLPSL implementation. The intruder (*i*) participates in the execution of protocol as a concrete session as shown in Fig 8. In the proposed protocol, we have three secrecy goals and three authentication goals. For example, the secrecy goal: secrecy of s1 indicates that the information ID_U, *b* and PW_U are kept secret to U_i only. The authentication goal: authentication_on ui_as_x denotes that the U_i has freshly generated random number *x* for the *AS*, where *x* is only known to U_i. When the *AS* receives *x* from messages of U_i, the *AS* checks a strong authentication for U_i based on *x*. Similarly, the other authentication goal authentication_on as_ui_n1 denotes that the *AS* generates a random number N_1 for U_i and when U_i receives N_1 from other messages from the *AS*, U_i checks a strong authentication for the *AS* based on N_1.

Analysis of simulation results. The proposed protocol is simulated under the widely-accepted OFMC and CL-AtSebackends using the SPAN, the Security Protocol ANimator for AVISPA [56]. Both back-ends are chosen for an execution test and a bounded number of sessions model checking [53]. Since the AVISPA implementation of our scheme in HLPSL uses bit XOR operation, currently SATMC and TA4SP backends do not support this feature. Due to this reason, the simulation results under both SATMC and TA4SP backends becomes inconclusive, and we have ignored these results in this paper.

The following verifications are performed in the proposed protocol as in [57]:

- *Executability check on non-trivial HLPSL specifications*: Due to some modelling mistakes, the protocol model sometimes cannot execute to completion. It may be then possible that the backends cannot find an attack, if the protocol model cannot reach a state where that attack can happen. Therefore, an executability test is very essential in AVISPA [54]. The executability check of the proposed protocol tells that the proposed protocol description is well matched with the designed goals as specified in Figs 5–9.
- *Replay attack check*: For replay attack check, the OFMC and CL-AtSe back-ends verify if the legitimate agents can execute the specified protocol by performing a search of a passive intruder. Both backends provide the intruder the knowledge of some normal sessions between the legitimate agents. The test results reported in Fig 10 clearly indicate that the proposed protocol is secure against the replay attack.
- *Dolev-Yao model check*: For the Dolev-Yao model check, the OFMC and CL-AtSe backends also check if there is any man-in-the-middle attacks possible by the intruder. In OFMC back-end, the depth for the search is nine and output of the results are shown in Fig 10. Also, the total number of nodes searched is 1040, which takes 2.56 seconds. On the other hand, in CL-AtSe backend, 63 states were analyzed and out of these states, all states were reachable. Further, CL-AtSe backend took 0.05 seconds for translation and 0.01 seconds for

```
role applicationserver (Ui, RS, AS : agent,
        % H is one−way hash function
        H: hash_func,
        SEND, RECV: channel(dy))

% Player: AS
played_by AS
def=
local State  : nat,
    IDu, B, PWu, USK, PSK, SIDs, P, Bu, Cus : text,
    Dus, Xs, Rs, Xu, SK, M2, N1, M3 : text,
    F : hash_func
const as_rs_xs, ui_as_xu, as_ui_n1,
    s1, s2, s3: protocol_id

init  State := 0
transition
% Application server registration phase

1. State = 0 /\ RECV(start) =|>
State' := 3 /\ Xs' := new() /\ Rs' := F(Xs'.P)
% Send registration request <SIDs, Rs > to RS publicly
        /\ SEND(SIDs.Rs')
% AS has freshly generated random value x_S for RS
        /\ witness(AS, RS, as_rs_xs, Xs')
% Login phase
% Receive message < AIDu, M1, Ru > from Ui publicly

2. State = 3 /\ RECV(xor(H(IDu.B'),F(Xu'.F(Xs'.P))).
            H(H(IDu.B').H(H(IDu.B').H(SIDs.PSK)).
            F(Xu'.P).F(Xu'.F(Xs'.P))).F(Xu'.P)) =|>
State' := 6 /\ secret({IDu,B',PWu}, s1, {Ui})
        /\ secret({USK,PSK}, s2, RS)
        /\ secret(Xs', s3, {RS, AS})
% Mutual authentication with key agreement phase
        /\ N1' := new()
        /\ SK' := H(F(Xu'.F(Xs'.P)).H(H(IDu.B').H(SIDs.PSK)).
                SIDs.N1')
        /\ M2' := xor(H(IDu.B'),N1')
        /\ M3' := H(SK'.H(IDu.B').N1'.H(H(IDu.B').H(SIDs.PSK)).
                F(Xu'.P))
% Send message < M2, M3 > to Ui publicly
        /\ SEND(M2'.M3')
% AS has freshly generated random value N1 for Ui
        /\ witness(AS, Ui, as_ui_n1, N1')
% Receive message < M4 > from Ui publicly

3. State = 6 /\
RECV(H(H(F(Xu'.F(Xs'.P)).H(H(IDu.B').H(SIDs.PSK)).
            SIDs.N1').N1')) =|>
% AS's acceptance of the value x_U generated for AS by Ui
State' :=  8 /\ request(Ui, AS, ui_as_xu, Xu')
end role
```

Fig 6. Role specification for application server *AS*.

doi:10.1371/journal.pone.0154308.g006

```
role registrationserver (Ui, RS, AS : agent,
        SKuirs : symmetric_key,
        % H is one−way hash function
        H: hash_func,
        SEND, RECV: channel(dy))
% Player: RS
played_by RS
def=
local State  : nat,
    IDu, B, PWu, USK, PSK, SIDs, P : text,
    Bu, Cus, Dus, Xs : text,
    F : hash_func
const as_rs_xs, s1, s2, s3: protocol_id
init  State := 0
transition
% User registration phase
% Receive request message < PIDu, Au > from Ui securely
1. State = 0 /\ RECV({H(IDu.B').H(PWu.B')}_SKuirs) =|>
        /\ secret({USK,PSK}, s2, RS)
        /\ Bu' := H(H(IDu.B').H(PWu.B').USK)
        /\ Cus' := H(H(IDu.B').H(SIDs.PSK))
        /\ Dus' := xor(H(PWu.B'),Cus')
% Send smart card SC to Ui securely
        /\ SEND({Bu'.SIDs.Dus'.H(SIDs.PSK).P.H}_SKuirs)
% Application server registration phase
% Receive registration request <SIDs, Rs > from AS publicly
2. State = 2 /\ RECV(SIDs.F(Xs'.P)) =|>
State' := 4  /\ secret(Xs', s3, {RS, AS})
% RS's acceptance of the value x_S generated for RS by AS
        /\ request(AS, RS, as_rs_xs, Xs')
end role
```

Fig 7. Role specification for registration server *RS*.

doi:10.1371/journal.pone.0154308.g007

computation. It is clear from the simulation results that the proposed protocol fulfills the design criteria and is secure under the test of AVISPA using OFMC and CL-AtSe backends with the bounded number of sessions.

Performance Analysis

This section demonstrates the performance analysis of the proposed protocol while considering various aspects such as security, computational cost and communication overhead. The performance analysis ensures that the proposed protocol is efficient and better in every aspect compared to Lu et al. [40] and other related protocols [2] [24] [25] [34] [39].

Functionality comparison

In this subsection, the proposed protocol is evaluated in terms of security and compared with other similar authentication protocols for multi-server architecture. The comparison of security properties between Chuang et al. [25], Mishra et al. [2], Lin et al. [39], Chen et al. [24], Lu et al. [40] and the proposed protocol are portrayed in Table 3. As shown in Table 3, the proposed protocol can withstand various security attacks and accomplishes distinct features such as user anonymity, untraceability, no verification tables, and biometrics deployment.

```
role environment()

def=
const ui, rs, as: agent,
skuirs: symmetric_key,
h, f : hash_func,
sids, p, ru, aidu, m1, m2, m3, m4 : text,
as_rs_xs, ui_as_xu, as_ui_n1,
s1, s2, s3: protocol_id
intruder_knowledge = {ui, rs, as, h, f, sids,
p, ru, aidu, m1, m2, m3, m4}

composition
session(ui, rs, as, skuirs, h)
/\ session(i, rs, as, skuirs, h)
/\ session(ui, i, as, skuirs, h)
/\ session(ui, rs, i, skuirs, h)
end role

goal
secrecy_of s1
secrecy_of s2
secrecy_of s3

authentication_on as_rs_xs
authentication_on ui_as_xu
authentication_on as_ui_n1

end goal

environment()
```

Fig 8. Role specification for the goal and environment.

doi:10.1371/journal.pone.0154308.g008

Computational cost comparison

It is evident from Table 4 that the computational cost of the proposed protocol is relatively lesser compared to Lu et al.'s and other similar protocols while accomplishing the significant security level as shown in Table 3. The proposed protocol is built on simple elliptic curve cryptography operations, one-way hash functions, concatenation and exclusive-OR operations. The computations of an exclusive-OR function and concatenation operation are relatively negligible, whereas exponential operation, elliptic curve point multiplication, encryption and

```
role session (Ui, RS, AS : agent,
        SKuirs : symmetric_key,
        % H is one−way hash function
        H: hash_func)
def=
  local  TX1, TX2, TX3, RX1, RX2, RX3 : channel
(dy)
composition
   user (Ui, RS, AS, SKuirs, H, TX1, RX1)
/\ registrationserver (Ui, RS, AS, SKuirs, H, TX2,
RX2)
/\ applicationserver (Ui, RS, AS, H, TX3, RX3)
end role
```

Fig 9. Role specification in HLPSL for the session.

doi:10.1371/journal.pone.0154308.g009

decryption operations consume quite more. Research has proven that there is always a trade-off between security and performance of a protocol. Usually, when the protocol becomes more secure, the computational cost becomes higher and vice versa. Contrarily, the proposed protocol succeeds to stabilize both the terms parallel. To evaluate the computational cost analysis, we give few notations for the involved actions in Chuang et al. [25], Mishra et al. [2], Lee et al. [34], Lin et al. [39], Chen et al. [24], Lu et al. [40] and the proposed protocol as shown below.

- T_h: Time complexity of a one-way hash function
- T_{mul}: Time complexity of a point multiplication operation on elliptic curve
- T_{fun}: Time complexity of encryption or decryption function
- T_c: Time for performing a chaotic map operation

```
% OFMC
% Version of 2006/02/13
SUMMARY
SAFE
DETAILS
BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
C:\progra~1\SPAN\testsuite
\results\auth.if
GOAL
as_specified
BACKEND
OFMC
COMMENTS
STATISTICS
parseTime: 0.00s
searchTime: 2.56s
visitedNodes: 1040 nodes
depth: 9 plies
```

```
SUMMARY
SAFE
DETAILS
BOUNDED_NUMBER_OF_SESSIONS
TYPED_MODEL
PROTOCOL
C:\progra~1\SPAN\testsuite
\results\auth.if

GOAL
As Specified
BACKEND
CL−AtSe
STATISTICS

Analysed : 63 states
Reachable : 63 states
Translation: 0.05 seconds
Computation: 0.01 seconds
```

Fig 10. The result of the analysis using OFMC and CL-AtSe backends.

doi:10.1371/journal.pone.0154308.g010

Table 3. Comparison of security properties.

Security property	Chuang [25]	Mishra [2]	Lin [39]	Chen [24]	Lu [40]	Our
P1	No	No	No	Yes	No	Yes
P2	Yes	Yes	No	Yes	Yes	Yes
P3	Yes	Yes	Yes	Yes	Yes	Yes
P4	No	No	Yes	Yes	No	Yes
P5	No	No	Yes	Yes	Yes	Yes
P6	No	No	Yes	Yes	No	Yes
P7	No	No	Yes	Yes	No	Yes
P8	Yes	Yes	No	Yes	Yes	Yes
P9	No	Yes	No	Yes	Yes	Yes
P10	Yes	Yes	Yes	Yes	Yes	Yes
P11	Yes	Yes	No	Yes	Yes	Yes
P12	Yes	Yes	No	Yes	No	Yes
P13	Yes	Yes	Yes	Yes	No	Yes

P1: User anonymity and untraceability, P2: Perfect mutual authentication, P3: Prevent replay attack, P4: Prevent man-in-middle attack, P5: Prevent stolen smart card attack, P6: Prevent user impersonation attack, P7: Prevent server impersonation attack, P8: Prevent insider attack, P9: Prevent denial-of-service attack, P10: Prevent password guessing attack, P11: No user verification table, P12: Prevent clock synchronization problem, P13: Perfect forward secrecy

doi:10.1371/journal.pone.0154308.t003

Table 4. Comparison of computational cost.

Phase	Chuang [25]	Mishra [2]	Lee [34]	Lin [39]	Chen [24]	Lu [40]	Our
Login	$3T_h$	$7T_h$	$2T_h+2T_c$	$5T_h+1T_{fun}$	$3T_h$	$5T_h$	$5T_h+2T_{mul}$
Authentication+key-agreement	$16T_h$	$17T_h$	$12T_h+4T_c$	$10T_h+4T_{mul}+5T_{fun}$	$16T_h$	$13T_h$	$8T_h+1T_{mul}$
Total	$19T_h$	$24T_h$	$14T_h+6T_c$	$15T_h+4T_{mul}+6T_{fun}$	$19T_h$	$18T_h$	$13T_h+3T_{mul}$

doi:10.1371/journal.pone.0154308.t004

Table 5. Comparison of communication overhead.

Feature	Chuang [25]	Mishra [2]	Lin [39]	Chen [24]	Lu [40]	Our
Number of messages	3	3	3	3	3	3
Number of bits	1280	1280	2528	1280	1216	960

doi:10.1371/journal.pone.0154308.t005

Communication overhead comparison

The communication overhead of the proposed protocol is compared with Chuang et al. [25], Mishra et al. [2], Lin et al. [39], Chen et al. [24], Lu et al. [40] and organized in Table 5. In order to evaluate the communication cost of the compared protocols, this paper considers SHA-1 hash function of 160 bits length, random number of 160 bits length, timestamp of 32 bits length, elliptic curve point of 160 bits length and 1024 bits modular prime for encryption and decryption function. As depicted in Table 4, the proposed protocol also uses 3 communication messages like the other similar protocols. In contrast, the proposed protocol requires only 960 bits for the 3 messages. Therefore, the proposed protocol consumes less bandwidth compared to Chuang et al. [25], Mishra et al. [2], Lin et al. [39], Chen et al. [24], Lu et al. [40] protocols.

Conclusions

This paper reviewed the recently proposed Lu et al.'s protocol for multi-server architecture and demonstrated that their protocol contains several weaknesses. In addition, this paper proposed an enhanced biometric based authentication with key-agreement protocol for multi-server architecture based on elliptic curve cryptography using smartcards. The mutual authentication of the proposed protocol is proved using BAN logic and also achieved significant features such as user anonymity, no verification tables, biometric authentication, perfect forward secrecy, with less computational and communication cost. The formal security of the proposed protocol is simulated and verified using the AVISPA tool to show that the proposed protocol can withstand active and passive attacks. The proposed protocol is perfectly suitable for practical applications as it accomplishes simple elliptic curve cryptography operations, one-way hash functions, concatenation operations and exclusive-OR operations. The formal and informal security analyses and performance analysis sections of this paper showed that the proposed protocol performs better in every aspect compared to Lu et al.'s protocol and existing similar protocols.

Author Contributions

Conceived and designed the experiments: AGR AKD VO KYY. Performed the experiments: AGR AKD VO KYY. Analyzed the data: AGR AKD VO KYY. Contributed reagents/materials/analysis tools: AGR AKD VO KYY. Wrote the paper: AGR AKD VO KYY.

References

1. Lamport L. (1981). Password authentication with insecure communication. *Communications of the ACM*, 24(11), 770–772. doi: 10.1145/358790.358797
2. Mishra D., Das A. K., & Mukhopadhyay S. (2014). A secure user anonymity-preserving biometric-based multi-server authenticated key agreement scheme using smart cards. *Expert Systems with Applications*, 41(18), 8129–8143. doi: 10.1016/j.eswa.2014.07.004
3. Li C. T., & Hwang M. S. (2010). An efficient biometrics-based remote user authentication protocol using smart cards. *Journal of Network and Computer Applications*, 33(1), 1–5. doi: 10.1016/j.jnca.2009.08.001
4. Amin R., Islam S. H., Biswas G. P., Khan M. K., & Kumar N. (2015). An efficient and practical smart card based anonymity preserving user authentication scheme for TMIS using elliptic curve cryptography. *Journal of medical systems*, 39(11), 1–18. doi: 10.1007/s10916-015-0351-y
5. Awasthi A. K., & Lal S. (2004). An enhanced remote user authentication protocol using smart cards. *Consumer Electronics*, *IEEE Transactions on*, 50(2), 583–586. doi: 10.1109/TCE.2004.1309430
6. Chien H. Y., Jan J. K., & Tseng Y. M. (2002). An efficient and practical solution to remote authentication: smart card. *Computers & Security*, 21(4), 372–375. doi: 10.1016/S0167-4048(02)00415-7
7. Das A. K. (2013). A secure and effective user authentication and privacy preserving protocol with smart cards for wireless communications. *Networking Science*, 2(1–2), 12–27. doi: 10.1007/s13119-012-0009-8
8. Fan C. I., & Lin Y. H. (2009). Provably secure remote truly three-factor authentication protocol with privacy protection on biometrics. *Information Forensics and Security*, *IEEE Transactions on*, 4(4), 933–945. doi: 10.1109/TIFS.2009.2031942
9. Goutham, R. A., Lee, G. J., & Yoo, K. Y. (2015). An anonymous ID-based remote mutual authentication with key agreement protocol on ECC using smart cards. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, 169–174. 10.1145/2695664.2695666.
10. Islam S. H., & Biswas G. P. (2012). An improved ID-based client authentication with key agreement scheme on ECC for mobile client-server environments. *Theoretical and Applied Informatics*, 24(4), 293. doi: 10.2478/v10179-012-0018-z
11. Islam S. K. (2014). Design and analysis of an improved smartcard-based remote user password authentication scheme. *International Journal of Communication Systems*. doi: 10.1002/dac.2793

12. Islam S. H., & Khan M. K. (2014). Cryptanalysis and improvement of authentication and key agreement protocols for telecare medicine information systems. *Journal of medical systems*, 38(10), 1–16. doi: 10.1007/s10916-014-0135-9
13. Islam S. H., & Biswas G. P. (2014). Dynamic id-based remote user mutual authentication scheme with smartcard using elliptic curve cryptography. *Journal of Electronics*, 31(5), 473–488. doi: 10.1007/s11767-014-4002-0
14. Islam S. H., Biswas G. P., & Choo K. K. R. (2014). Cryptanalysis of an improved smartcard-based remote password authentication scheme. *Information Sciences Letters*, 3(1), 35.
15. Islam S. H., Khan M. K., Obaidat M. S., & Muhaya F. T. B. (2015). Provably secure and anonymous password authentication protocol for roaming service in global mobility networks using extended chaotic maps. *Wireless Personal Communications*, 84(3), 2013–2034. doi: 10.1007/s11277-015-2542-8
16. Islam S. H., & Biswas G. P. (2015). Cryptanalysis and improvement of a password-based user authentication scheme for the integrated EPR information system. *Journal of King Saud University-Computer and Information Sciences*, 27(2), 211–221. doi: 10.1016/j.jksuci.2014.03.018
17. Islam S. H., Das A. K., & Khan M. K. (2015) "A novel biometric-based password authentication scheme for client-server environment using ECC and fuzzy extractor," *International Journal of Ad Hoc and Ubiquitous Computing*, In Press.
18. Islam SH, Khan MK, Li X (2015) Security Analysis and Improvement of 'a More Secure Anonymous User Authentication Scheme for the Integrated EPR Information System'. PLoS ONE 10(8): e0131368. doi: 10.1371/journal.pone.0131368 PMID: 26263401
19. Lee J. K., Ryu S. R., & Yoo K. Y. (2002). Fingerprint-based remote user authentication protocol using smart cards. *Electronics Letters*, 38(12), 554–555. doi: 10.1049/el:20020380
20. Lin C. H., & Lai Y. Y. (2004). A flexible biometrics remote user authentication protocol. *Computer Standards & Interfaces*, 27(1), 19–23.
21. Mir O., van der Weide T., & Lee C. C. (2015). A secure user anonymity and authentication scheme using AVISPA for telecare medical information systems. *Journal of Medical Systems*, 39(9), 1–16. doi: 10.1007/s10916-015-0265-8
22. Song R. (2010). Advanced smart card based password authentication protocol. *Computer Standards & Interfaces*, 32(5), 321–325. doi: 10.1016/j.csi.2010.03.008
23. Yang W. H., & Shieh S. P. (1999). Password authentication protocols with smart cards. *Computers & Security*, 18(8), 727–733. doi: 10.1016/S0167-4048(99)80136-9
24. Chen C. T., & Lee C. C. (2015). A two-factor authentication scheme with anonymity for multi-server environments. *Security and Communication Networks*, 8(8), 1608–1625. doi: 10.1002/sec.1109
25. Chuang M.-C., & Chen M. C. (2014). An anonymous multi-server authenticated key agreement scheme based on trust computing using smart cards and biometrics. *Expert Systems with Applications*, 41(4), 1411–1418. doi: 10.1016/j.eswa.2013.08.040
26. Das A. K., Odelu V., & Goswami A. (2015). A Secure and Robust User Authenticated Key Agreement Scheme for Hierarchical Multi-medical Server Environment in TMIS. *Journal of Medical Systems*, 39 (9), 1–24. doi: 10.1007/s10916-015-0276-5
27. Guo D. L., & Wen F. T. (2014). Analysis and improvement of a robust smart card based-authentication scheme for multi-server architecture. *Wireless Personal Communications*, 78(1), 475–490. doi: 10.1007/s11277-014-1762-7
28. Hsiang H. C., & Shih W. K. (2009). Improvement of the secure dynamic ID based remote user authentication protocol for multi-server environment. *Computer Standards & Interfaces*, 31(6), 1118–1123. doi: 10.1016/j.csi.2008.11.002
29. Huang C. H., Chou J. S., Chen Y., & Wun S. Y. (2012). Improved multi-server authentication protocol. *Security and Communication Networks*, 5(3), 331–341. doi: 10.1002/sec.332
30. Islam S. H. (2014). A provably secure ID-based mutual authentication and key agreement scheme for mobile multi-server environment without ESL attack. *Wireless Personal Communications*, 79(3), 1975–1991. doi: 10.1007/s11277-014-1968-8
31. Juang W. S. (2004). Efficient multi-server password authenticated key agreement using smart cards. *Consumer Electronics*, *IEEE Transactions on*, 50(1), 251–255. doi: 10.1109/TCE.2004.1277870
32. Lee C. C., Lin T. H., & Chang R. X. (2011). A secure dynamic ID based remote user authentication protocol for multi-server environment using smart cards. *Expert Systems with Applications*, 38(11), 13863–13870. doi: 10.1016/j.eswa.2011.04.190
33. Lee C. C., Lai Y. M., & Li C. T. (2012). An improved secure dynamic ID based remote user authentication scheme for multi-server environment. *International Journal of Security and Its Applications*, 6(2), 203–209.

34. Lee C. C., Lou D. C., Li C. T., & Hsu C. W. (2014). An extended chaotic-maps-based protocol with key agreement for multiserver environments. *Nonlinear Dynamics*, 76(1), 853–866. doi: 10.1007/s11071-013-1174-3

35. Li X., Xiong Y., Ma J., & Wang W. (2012). An efficient and security dynamic identity based authentication protocol for multi-server architecture using smart cards. *Journal of Network and Computer Applications*, 35(2), 763–769. doi: 10.1016/j.jnca.2011.11.009

36. Li X., Ma J., Wang W., Xiong Y., & Zhang J. (2013). A novel smart card and dynamic ID based remote user authentication scheme for multi-server environments. *Mathematical and Computer Modelling*, 58 (1), 85–95. doi: 10.1016/j.mcm.2012.06.033

37. Li C. T., Lee C. C., Weng C. Y., Fa C. I. (2015). "A Secure Dynamic Identity based Authentication Protocol with Smart Cards for Multi-Server Architecture," *Journal of Information Science and Engineering*, 31(6), 1975–1992.

38. Liao Y. P., & Wang S. S. (2009). A secure dynamic ID based remote user authentication protocol for multi-server environment. *Computer Standards & Interfaces*, 31(1), 24–29. doi: 10.1016/j.csi.2007.10.007

39. Lin H., Wen F., & Du C. (2015). An Improved Anonymous Multi-Server Authenticated Key Agreement Scheme Using Smart Cards and Biometrics. *Wireless Personal Communications*, 1–12. doi: 10.1007/s11277-015-2708-4

40. Lu Y, Li L, Yang X, Yang Y (2015) Robust Biometrics Based Authentication and Key Agreement Scheme for Multi-Server Environments Using Smart Cards. *PLoS ONE*, 10(5): e0126323. doi: 10.1371/journal.pone.0126323 PMID: 25978373

41. Odelu V., Das A. K., & Goswami A. (2015). A Secure Biometrics-Based Multi-Server Authentication Protocol Using Smart Cards. *Information Forensics and Security*, *IEEE Transactions on*, 10(9), 1953–1966. doi: 10.1109/TIFS.2015.2439964

42. Sood S. K., Sarje A. K., & Singh K. (2011). A secure dynamic identity based authentication protocol for multi-server architecture. *Journal of Network and Computer Applications*, 34(2), 609–618. doi: 10.1016/j.jnca.2010.11.011

43. Tsai J. L. (2008). Efficient multi-server authentication protocol based on one-way hash function without verification table. *Computers & Security*, 27(3), 115–121. doi: 10.1016/j.cose.2008.04.001

44. Wang R. C., Juang W. S., & Lei C. L. (2009). User authentication protocol with privacy-preservation for multi-server environment. *Communications Letters*, *IEEE*, 13(2), 157–159. doi: 10.1109/LCOMM.2009.081884

45. Xue K., Hong P., & Ma C. (2014). A lightweight dynamic pseudonym identity based authentication and key agreement protocol without verification tables for multi-server architecture. *Journal of Computer and System Sciences*, 80(1), 195–206. doi: 10.1016/j.jcss.2013.07.004

46. Yoon E. J., & Yoo K. Y. (2013). Robust biometrics-based multi-server authentication with key agreement scheme for smart cards on elliptic curve cryptosystem. *The Journal of Supercomputing*, 63(1), 235–255. doi: 10.1007/s11227-010-0512-1

47. Burrows M., Abadi M., & Needham R. M. (1989, December). A logic of authentication. In *Proceedings of the Royal Society of London A*: *Mathematical*, *Physical and Engineering Sciences*. The Royal Society (Vol. 426, No. 1871, pp. 233–271). doi: 10.1098/rspa.1989.0125

48. AVISPA. Automated Validation of Internet Security Protocols and Applications. http://www.avispaproject.org/. Accessed on October 2015.

49. Kocher P., Jaffe J., & Jun B. (1999b). Differential power analysis. *In Proceedings of advances in cryptology—CRYPTO'99*. LNCS (Vol. 1666, pp. 388–397). doi: 10.1007/3-540-48405-1_25

50. Messerges T. S., Dabbish E. A., & Sloan R. H. (2002b). Examining smart-card security under the threat of power analysis attacks. *IEEE Transactions on Computers*, 51(5), 541–552. doi: 10.1109/TC.2002.1004593

51. Nam J, Choo K-KR, Han S, Kim M, Paik J, Won D (2015) Efficient and Anonymous Two-Factor User Authentication in Wireless Sensor Networks: Achieving User Anonymity with Lightweight Sensor Computation. *PLoS ONE*, 10(4): e0116709. doi: 10.1371/journal.pone.0116709 PMID: 25849359

52. Armando et al. (2005). The AVISPA Tool for the Automated Validation of Internet Security protocols and Applications. In P*roc. of International Conference on Computer Aided Verification* (CAV'05), Scotland, UK, vol. 3576, pp. 281–285. 10.1007/11513988_27

53. Basin D., Mödersheim S., & Luca V.. (2005). OFMC: A symbolic model checker for security protocols. *International Journal of Information Security*, 4(3), 181–208. doi: 10.1007/s10207-004-0055-7

54. Von Oheimb, D. (2005, September). The high-level protocol specification language HLPSL developed in the EU project AVISPA. In *Proceedings of APPSEM 2005 workshop* (pp. 1–17).

55. Dolev D., & Yao A. C. (1983). On the security of public key protocols. *Information Theory*, *IEEE Transactions on*, 29(2), 198–208. doi: 10.1109/TIT.1983.1056650

56. AVISPA. SPAN, the Security Protocol Animator for AVISPA. http://www.avispa-project.org/. Accessed on December 2015.

57. Lv C., Ma M., Li H., Ma J., & Zhang Y. (2013). A novel three-party authenticated key exchange protocol using one-time key. *Journal of Network and Computer Applications*, 36(1), 498–503. doi: 10.1016/j.jnca.2012.04.006