International Journal of Information Security manuscript No.
(will be inserted by the editor)

Scalable, password-based and threshold authentication for Smart Homes*

Andrea Huszti [a,1], **Szabolcs Kovács**[b,2], **Norbert Oláh**[c,3]

[1]Faculty of Informatics, University of Debrecen, Debrecen, Hungary
[2]CCLab Ltd., Faculty of Informatics, University of Debrecen, Debrecen, Hungary
[3]Faculty of Informatics, University of Debrecen, Debrecen, Hungary

Received: date / Accepted: date

Abstract Smart homes are a special use-case of the IoT paradigm, which is becoming more and more important in our lives. Although sensors, devices and applications make our daily lives easier, they often collect our sensitive data, which may lead to security problems (e.g., hacked devices, botnets, etc.). In several cases the appropriate security mechanisms are missing within the devices. Therefore, security measures have become a central topic in the field of IoT. The most essential requirements are secure user-device authentication and confidentiality of transferred sensitive data.

Passwords are the most widely used factors in various areas, such as user authentication, key establishment, and also secret sharing. Password-based protocols that are resistant to typical threats, such as offline dictionary, man-in-the-middle and phishing attacks generate new session keys. The major aim of these solutions is to guarantee high level security, even if a user applies a single low-entropy human memorable password for all their accounts.

We introduce a threshold and password-based, distributed, mutual authenticated key agreement with key confirmation protocol for a smart home environment. The proposed protocol is a scalable and robust scheme, which forces the adversary to corrupt $l-1$ smart home devices, where l is the threshold, in order to perform an offline dictionary attack. The protocol is designed to achieve password-only setting, and end-to-end security if the chosen IoT devices are also authenticated besides the user. We also provide a security analysis of the protocol in AVISPA. We apply the On-the-fly Model-Checker and the Constraint-Logic-Based Attack Searcher to perform protocol verification for bounded numbers of sessions. We show that the proposed protocol provides session key secrecy and mutual authentication of the user and the device manager. Since efficiency is a crucial aspect, we implemented our protocol to measure the computation and communication costs, and demonstrate that our solution is appropriate and eligible for smart homes.

1 Introduction

Nowadays, the Internet of Things (IoT) is becoming more and more important in our lives. Smart homes are a special use-case of the IoT paradigm. Users control their home's electronic systems (smart bulbs, smart locks, sensors etc.) by using smartphone applications and even from remote locations. In smart homes large variety of devices interact over the local network using different communication technologies (e.g., Wi-Fi, Bluetooth, Zigbee, Z-Wave, LoRa, cellular network, etc.) to allow data flow between other devices [52]. Zigbee or Z-Wave IoT devices are usually connected to a smart home hub constructing a mesh network [52] and access LAN through the hub. There are IoT devices that are connected directly to the home Internet router and communicate with the smart home hub via the LAN or the cloud.

Smart home systems also have other unique features that place additional design requirements. Since most devices have limited computing resources and lack or have minimal security functionalities (no authentication and plain text data transmission) several recommendations are available to handle these aspects, such as Software Defined Networking (SDN), which offers programmability, agility and centralized management. In [41] a privacy preserving communication scheme is proposed for SDN enabled smart homes (PCSS), which aims at provisioning user and smart device

*The presented research has been supported by the SETIT Project (no. 2018-1.2.1-NKP-2018-00004), which has been implemented with the support provided by the National Research, Development and Innovation Fund of Hungary, financed under the 2018-1.2.1-NKP funding scheme.

[a]e-mail: huszti.andrea@inf.unideb.hu
[b]e-mail: szabolcs.kovacs@cclab.com, kovacs.szabolcs@inf.unideb.hu
[c]e-mail: olah.norbert@inf.unideb.hu

authentication, privacy for data and user queries. In most cases there is a central node or a smart home hub which is responsible for managing communications with the other nodes of the local network and outside. This proposition applies lightweight cryptography (xor, hash function, AES) for the authentication between the user and the smart device with the use of a central controller, focuses on data storage and provides privacy to the searchable encrypted user queries. Moreover, this scheme does not use the distributed properties of the smart home and addresses other issues that we consider. Smart devices send sensitive data via wireless connection to the central node, where immediate analysis and decision are performed to send back control commands ([49], [64]). In smart homes EdgeOS_H [21] and HomePad [69] are two solutions to address edge-based IoT issues.

In Wireless physical communication attackers can intercept communications more easily, channels may reveal sensitive information regarding user interaction, behavior, lifestyle or physical activity. IoT devices are often very vulnerable due to weak protection (weak or default passwords) and poor maintenance. Numerous studies have addressed the security vulnerabilities of IoT devices ([45], [44], [62]). Bugs have been found in a wide range of devices including routers [67], smartcams [60], baby monitors [13] and smart plugs [50]. However there are many propositions and ideas which try to fix these vulnerabilities.

Recently several security vulnerabilities are exploited in smart home hub devices. The smart home hub is an appealing target for cybercriminals that can serve as an entry-point for remote attacks. In July 2019, researchers at BlackMarble found weakness with ZipatoâĂŹs ZipaMicro smart hubs where exploiting Pass-the-Hash security flaws an adversary could open a smart lock connected to the hub [47]. ESET IoT Research has also found several vulnerabilities in three different hubs [31]. Attackers could gain hardcoded passwords or could change passwords.

Only those with the appropriate access control should be able to retrieve confidential data, which increases the role of authentication.

1.1 Motivation and related works

Karim Arabi [2] defined edge computing broadly as all computing outside the cloud happening at the edge of the network, and more specifically in applications where real-time processing of data is required. The devices in the smart homes collect and transmit data to the edge who pre-processes the data and transmits it to the cloud. Some device data, such as camera shots, are also available locally on the edge for a while. The edge tool can not only process data from sensors but also give it certain instructions. Therefore, establishing a proper authentication between the user and the edge device is of paramount importance. After authentication, a session key agreement must be provided between the user and the edge device because the user can request confidential information or instruct the edge.

In the case of *key agreement*, both entities contribute to the joint secret key by providing information from which the key is derived. In *mutual authentication* parties who engage in a conversation always make sure that it is the other with whom he speaks. In protocols providing *implicit key authentication*, each participant is assured that only the intended parties can learn the value of the session key. A key agreement protocol that provides mutual implicit key authentication is called an *authenticated key agreement* protocol (or AK protocol). A key agreement protocol provides key confirmation between two participants where the participants make sure that the other participant possesses the secret key. A protocol that provides mutual key authentication as well as mutual key confirmation is called an *authenticated key agreement with key confirmation* protocol (or an AKC protocol).

In scientific literature wide variety of authentication schemes can be found, where the mutual authentication is processed between a user and a service provider consisting of several devices, servers or participants. Usually centralized, one-factor [37, 46] or two-factor identity verification protocols [22, 28] are proposed. Xavier Boyen applies blind signatures for a Hidden Credential Retrieval protocol [18] that uses a user password. Saeed et. al. [59] recommended an authenticated key agreement for IoTs (AKAIoTs). The proposed AKAIoTs can be used to establish a secure shared key between WSNs client and cloud server. The shared key can be used for secure data transmission between WSNs client and cloud server. The concept of distributed authentication of multiple devices (sensors, servers, etc.) enhances the security level, since multiple devices simultaneously are have to be broken. Multi-factor authentication can also provide an enhanced level of assurance in higher-security scenarios because a multi-factor protocol is designed to remain secure even if all but one of the factors has been compromised.

Brainard et.al. suggested a two-server approach in [20], where two servers together decide on the correctness of the password submitted for authentication as well. Katz et. al. [43] demonstrated a two-server protocol with a password-only setting, in which the user needs to remember only the password, and not the servers' public keys. In Acar et. al. solution [1] the secret key is securely stored and blinded by some function of user password at storage provider(s) different from the login server. In this paper [48] a threshold MFA key exchange protocol is recommended which is built on the top of a threshold oblivious pseudorandom function and an authenticated key exchange protocol. The authors can directly assume an authenticated and secure channel between clients and devices and typically, smart home environments are not fit for this assumption.

Jian Shen et.al. [61] proposed a one-to-many group authentication protocol with a group key establishment between personal digital assistance and each sensor node. They demonstrated a certificateless authentication protocol without pairings based on certificateless cryptography between PDA and application provider, using ECC algorithms. In [36], a multi-server authentication protocol is designed, where one-time passwords are shared among the cloud servers. A Merkle tree or a hash tree [54] is applied for verifying the correctness of the one-time password. A variant of authenticated key exchange is demonstrated in [68], which includes a permanent Control Server and cloud servers on 5G network. Our solution differs from these papers since we can scale the generated long-lived keys on the user's and the provider's sides as well.

A novel security protocol is introduced by Mazhar Rathore et.al. [58], which simplifies the mutual authentication and key exchange among smart home devices. The protocol leverages identity-based cryptography (IBC), thus alleviating the requirement of storing and managing public key certificates. The protocol maintains the security of the honest devices' private keys, even when the admin device is compromised. They used a secret sharing technique for the new device registration and applied bilinear pairing in the authentication phase. In our solution the advantages of Mazhar's protocol are provided. Moreover, we formalize the security analysis of our protocol via AVISPA tool and in our protocol the devices do not have to compute the bilinear pairing, which would be costly for the devices. A cross-platform identity-based system using Webassembly is suggested in [66]. They introduced a new library, called CryptID, which is an open-source IBC implementation for desktops, mobiles, and IoT platforms.

Nowadays, password is an important factor in user authentication. There are several suggestions for implementing password-authenticated key exchange (PAKE), which was proposed first by Bellovin and Merritt [12]. Passwords have a number of vulnerabilities which can be reduced in several ways. In [17] a Diffie-Hellman-based password-authenticated key exchange protocol is proposed. Their protocol provides a formal proof of security in the random oracle model against the passive and active adversaries. A number of two-factor authentication schemes have been proposed that depend on a long cryptographic secret and a short password ([55, 65]). Furthermore, forward secrecy is crucial and appears in several schemes ([34, 35, 56]). Pointcheval and Zimmer ([56]) demonstrated a multi-factor authentication scheme applying a password, a long cryptographic secret, and biometric data as well. Douglas Stebila et.al. [29] took a form of authenticated key exchange into consideration which is based on multi-factor password authenticated key exchange. The session establishment depends on successful authentication of multiple short secrets, which are based on a long-term password and a one-time response. However, there would still be a problem with the required additional piece of biometric data or other equipments (e.g., smart card), which would bring burden to users.

	T, SS	OF	SCA	EA	SA	I
MacKenzie et. al.	x			x	x	
Raimondo, Gennaro	x					
Bagherzandi et. al.	x			x	x	
Jarecki et. al.	x			x	x	
Işler, Küpçü - TSPA	x	x		x	x	x
Işler, Küpçü - DSPP	x			x		x
Our	x	x	x	x	x	x

Table 1 Summary table of threshold password-based authenticated key exchange solutions. T: Threshold, S: Secret Sharing, OF: Other factor besides the password, SCA: Scalability, EA: Efficient Analysis, SA: Security Analysis, I: Implementation

A threshold variant of PAKE is recommended in [51]. They apply multiple servers in a threshold scheme to prevent the threat of server corruption and handle password vulnerability. These solutions expect that treshold-number servers may become corrupt so not all servers are colluding. Most of these propositions are not forward secure or perfect forward secure. Two threshold password authenticated key exchange schemes are proposed by Mario Di Raimondo and Rosario Gennaro[57], where the protocols require $n > 3t$ servers, where n denotes the number of the servers and t is the threshold. They enforce the transparency property: from the client's point of view the protocol should look exactly like the centralized protocol of Katz, Ostrovsky, and Yung (KOY protocol) [43]. They provide formal security analysis of the protocol but the scheme is not a suitable choice for IoT or smart home environment since it applies fifteen exponentations and uses digital signatures that are resource-intensive tasks.

A threshold Password-Protected Secret Sharing (PPSS) scheme was formalized by Bagherzandi et. al. [8]. Jarecki et. al. [42] present the first round-optimal PPSS scheme, requiring just one message from user to server and from server to user, and prove its security in the challenging password-only setting where users do not have access to an authenticated public key. Although it isn't scalable, they demonstrate that their protocol is very efficient. These recommendations [38, 40] consider similar contributions to our proposition (scalability, robustness, password usage, etc.). These solutions are more suitable in the cloud environment and their protocols contain storage providers. However, our solution is recommended typically for a smart home environment.

The comparison of the existing Threshold Password-based Authenticated Key Exchange solutions is demonstrated with various attributes in Table 1.

1.2 Our contribution

Our goal is to propose a password-based multi-device authentication scheme for smart homes to reduce security vulnerabilities.

The proposed protocol is designed typically for smart home environment (Figure 1). There are several IoT devices and at least one central node or edge, therefore instead of centralized authentication (e.g., Kerberos) we propose a multi-device authentication. The central node or edge is called the *device manager*. If one or more devices break down or become compromised, the system will still be able to authenticate the user in a secure way. Hence we thoroughly utilize the capabilities of these systems like *robustness* and *greater availability*.

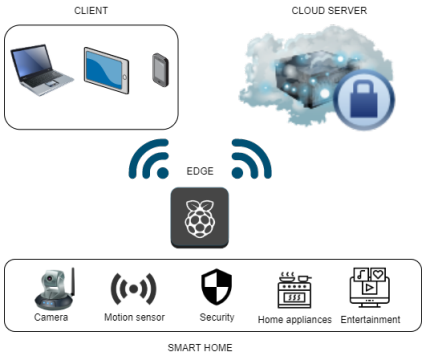

Fig. 1 Smart Home

The scheme is an authenticated key exchange protocol with key confirmation (AKC) which takes advantage of the distributed IoT system. The client's password is shared among the smart home devices. Thus, several sensors and devices together verify the correctness of the user password. Attackers need to attack multiple devices simultaneously in order to impersonate the user successfully. Distributed storage of the passwords provides resistance against offline attacks as well. We accomplish the *password-only setting*, hence a user needs to know only a password. Since smart home devices (e.g., cameras) generate a lot of sensitive data, *confidentiality of data* needs to be ensured during the communication between the parties, and besides the identity verification of the user and the smart home a session key is also generated.

Our scheme is designed to be an AKC between the user and the device manager. However if the user would like to connect to an IoT device directly, the proposed protocol provides end-to-end security as well.

Since there are resource constrained devices, we put a great emphasis on *efficiency* during the design. The session key is generated by Elliptic Curve Diffie-Hellman key exchange, moreover hash, MAC, xor operations and raising to a power are applied.

The protocol consists of two phases: setup and authentication. During the setup phase the password is chosen by the user and split among the devices. Whenever the user logs in to the smart home system the authentication phase is run, the password given by the user is verified by the devices.

The number of devices in a smart home system is changing. The more devices there are, the higher security the system should provide, *i.e.,* the password verification should be processed by more devices. The proposed protocol is *scalable* in an efficient way. The password is not necessarily changed every time the number of devices is increased, hence the shares already set for the installed devices are not changed. For the newly registered devices new shares of the same password are set.

To provide higher security level, during the authentication phase the user chooses a random value called *authentication value*, which is securely split among the devices. For the authentication value a threshold based on the number of devices, called *authentication threshold* is set. At least threshold number of devices are necessary to construct the authentication value from its shares. The authentication threshold is greater than or equal to the password threshold. A device calculates its authentication share with the help of its symmetric key based on its password share. Consequently, the smart home system authenticates the user successfully only if the authentication value is calculated, *i.e.,* only if at least authentication threshold number of devices participate. Increasing the number of devices results in a larger authentication threshold, hence greater security level is achieved. Similarly, the number of devices can be decreased. Reregistration is required only if it goes below the threshold of the password secret sharing.

The smart home is also authenticated. The user verifies whether the devices are able to correctly calculate the password and the authentication value. Valid verification value is constructed only if the devices possess valid password shares.

Secret sharing algorithm and bilinear map are adopted to provide resistance against offline attacks. To construct the password at least threshold number of shares are necessary. Let l denote the password threshold and assume that $l-1$ devices are compromitted. With the knowledge of $l-1$ password shares the adversary can launch a dictionary attack. For each possible password the authentication value should be constructed from its shares for the verification. Besides the hash, these calculations are also required for each dictionary element that slows down the attack. We apply a bilinear map for storing the hash value of the password and the salt. A hash and a bilinear map calculation together should be carried out for each possible password that also slows down the attack.

We have validated the security properties of the proposed protocol by using an automated, security protocol val-

idation tool, named AVISPA. We show that our protocol provides mutual authentication of the participants and the generated fresh session key is kept secret. We formalize the protocol in HLPSL and also define the above security goals for the analysis. AVISPA supports four types of goal predicates:witness (for weak authentication), request (for strong authentication), and secret.

1.3 Outline of article

In Section 2, the necessary preliminaries are detailed. In Section 3, the steps of the protocol are described in detail. In Section 4, we provide a security analysis, which contains the description of AVISPA with the security requirements. We also formalize the protocol in HLSPL, and introduce the security goals. In Section 5 and 6 practical issues and the conclusion are presented.

2 Preliminaries

We review some of the theory of elliptic curves over finite fields, further details can be found in [53]. Let $\mathbb{F}_q$ denote the finite field containing q elements. We apply E to define elliptic curve over $\mathbb{F}_q$, where $q = p^m$ and p is the characteristic of $\mathbb{F}_q$. If $p > 3$, then $E(\mathbb{F}_q)$ is the set of all solutions in $\mathbb{F}_q \times \mathbb{F}_q$ to an affine equation

$$y^2 = x^3 + ax + b,$$

with $a, b \in \mathbb{F}_q$ and the $4a^3 + 27b^2 \neq 0$, together with an extra point O, called the point at infinity. The main advantage of applying elliptic curves in cryptography is that the same level of security is ensured with shorter key lengths comparing to other public-key systems. Hence elliptic curve cryptographic primitives work on most of the resource-constrained devices.

Elliptic Curve Cryptography (ECC) is based on the infeasibility of Elliptic Curve Discrete Logarithm Problem, where the definition is the following:

Definition 1 Let $G \in E(\mathbb{F}_q)$ be a point of order n, and let $< G >$ be the subgroup of $E(\mathbb{F}_q)$ generated by G. The Elliptic Curve Discrete Logarithm Problem is to determine the value of $a \in \mathbb{Z}_n$ in the equation $A = aG$, for a given point $A \in < G >$.

One of the most often used key agreement protocol is the Diffie-Hellman protocol. The purpose of the Diffie-Hellman protocol is to enable two parties to securely exchange a session key which can then be used for encrypting and authenticating messages. The security of the Elliptic Curve Diffie-Hellman protocol is based on the infeasibility of calculating elliptic curve discrete logarithms. The algorithm works as follows:

- Alice and Bob select random secret values a, $b \in \mathbb{Z}_q^*$, respectively.
- Alice and Bob calculate aG and bG, respectively, where G is a generator in $E(\mathbb{F}_q)$ and send their values to each other.
- Both of them compute the shared secret $K = a(bG) = a(bG) = abG$.

For storing password information on the devices *bilinear pairings* are applied. Let us review the definition of the admissible bilinear map [19].

Definition 2 Let G_1 and G_2 be two groups of order q for some large prime q. A map $\hat{e} : G_1 \times G_1 \to G_2$ is an admissible bilinear map if satisfies the following properties:

1. Bilinear: We say that a map $\hat{e} : G_1 \times G_1 \to G_2$ is bilinear if $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$ for all $P, Q \in G_1$ and all $a, b \in \mathbb{Z}$.
2. Non-degenerate: The map does not send all pairs in $G_1 \times G_1$ to the identity in G_2. Since G_1, G_2 are groups of prime order, if P is a generator of G_1 then $\hat{e}(P, P)$ is a generator of G_2.
3. Computable: There is an efficient algorithm to compute $\hat{e}(P, Q)$ for any $P, Q \in G_1$.

It is important to note that bilinearity can be restated for all $P, Q, R \in G_1$ $\hat{e}(P + Q, R) = \hat{e}(P, R)\hat{e}(Q, R)$ and $\hat{e}(P, Q + R) = \hat{e}(P, Q)\hat{e}(P, R)$. We can find G_1 and G_2 where these properties hold. The Weil and Tate pairings prove the existence of such constructions. Typically, G_1 is an elliptic-curve group and G_2 is the multiplicative group of a finite field. Bilinear map is chosen due to its one-wayness and bilinear properties.

The Bilinear Diffie-Hellman Problem is strongly related to the bilinear map.

Definition 3 Let $\hat{e} : G_1 \times G_1 \to G_2$ be a bilinear map on (G_1, G_2) and $a, b, c \in Z_q^*$. Given (P, aP, bP, cP) and compute $\hat{e}(P, P)^{abc}$.

Using a (k, n) threshold scheme a secret S can be divided into n shares in a way that $k \leq n$ will be the threshold number of the shares we must be able to compute S. Thus $k - 1$ or fewer shares leave S completely undetermined. We apply secret sharing for the IoT devices to construct the password.

Shamir's Secret Sharing threshold scheme is based on polynomial interpolation. We need at least k points to define a polynomial of degree $k - 1$. To divide a secret S into shares a polynomial degree $k - 1$ is set: $f(x) = a_1 + a_1 x + ... + a_{k-1} x^{k-1}$, where $a_i \in \mathbb{Z}_q$, where q is a large prime and $i = 1, \ldots, k - 1$ are randomly chosen. Secret $S = f(0) = a_1$ and the shares will be: $s_1 = f(1), \ldots, s_n = f(n)$.

To find the secret S Lagrange polynomials are applied. There is an optimized approach, where

$$S = f(0) = \sum_{j=0}^{k-1} f(x_j) \prod_{\substack{m=0 \\ m \neq j}}^{k-1} \frac{x_m}{x_m - x_j}.$$

3 The proposed scheme

In this section an AKC protocol is proposed that is carried out between a user and a smart home environment including a device manager connected to the IoT devices. During the *setup phase* the user sets up his smart home system, chooses a password that is split into shares. The shares are securely stored on each device including the device manager. From the stored value a password share-based long-lived symmetric secret key K_i is calculated, where $i = 1, \ldots, n$, where n is the number of devices.

A share-based long-lived key K_i depends on a secret share of the password and the salt value. The shares are static until the password is valid. Whenever the password is changed new shares are generated. If the number of devices is increased, the user with the help of the setup module sets a new share of the same password for the new device. The stored values, hence the long-lived keys of the other IoT devices that are already installed are not modified, *i.e.,* by increasing the number of devices the password threshold is still the same.

In the authentication phase mutual authentication of the user and the smart home system is processed. The authentication is run by the user and m randomly chosen devices, where $k \leq m \leq n$ (k is the authentication threshold, *i.e.,* at least k devices are necessary for calculating the authentication value). The authentication threshold is based on the number of devices and set by the device manager. Hence by increasing the number of devices the authentication threshold is also increased. In case the number is decreased, the authentication threshold can also be decreased. The authentication threshold is greater than or equal to the password threshold.

At the beginning of the authentication phase short-lived symmetric keys ($\overline{K}_i$) are exchanged securely between the edge and each device, moreover the user chooses a *random authentication* value that is split among all the devices. The secret shares of the authentication value are sent securely to the edge that randomly chooses m devices, where m is greater than or equal to the authentication threshold. The edge transmits the authentication shares securely to the devices, which are able to calculate the authentication share only with the password share-based long-lived keys. The m devices together calculate a value which depends on the user's password, the salt and the chosen authentication value and send it to the user. If this value is correct, devices prove the knowledge of the password share-based long-lived keys, hence the smart home system is authenticated. Moreover, the hash of this value is also transmitted to the edge. If it is correct, then the password and the salt are also valid, hence the user is authenticated as well.

In addition to mutual authentication of the participants, a secret session key is also exchanged between the user and the device manager applying Elliptic Curve Diffie-Hellman key exchange.

It is assumed that a client software is running on the client device (e.g., laptop, mobile phone etc.) that requires the password from the user to initiate the authentication process. Having the password, the share-based long-lived keys K_i are calculated with the help of a salt value stred by the client sofware and the execution of authentication begins.

3.1 Setup phase

There are two participants in our protocol. One of them is the *IoT system* including the manager device and the IoT devices $(J_1, \ldots, J_n)$ and the other one is the *user* (I), which queries services and data. The set of all binary strings of arbitrary length is denoted by $\{0,1\}^*$. If x, y are strings, then $x||y$ denotes the concatenation of x and y. Let $\oplus$ denote an exclusive or calculation. During setup, all system parameters and long-lived keys are generated. Let E denote an elliptic curve defined over a finite field $\mathbb{F}$ and $G \in E(\mathbb{F})$ be a point of order $\mathbf{n}$. Elliptic curve parameters are chosen in a way that the system resists all known attacks on the elliptic curve discrete logarithm problem in $< G >$. Let G_1 and G_2 be two groups of order q for some large prime q, map $\hat{\hat{e}} : G_1 \times G_1 \to G_2$ be an admissible bilinear map. System parameters *par* are given by $par = (E, \mathbb{F}, \mathbf{n}, G, G_1, G_2, \hat{e}, H, H_0, H_1, Mac)$, where $H : \{0,1\}^* \to \{0,1\}^{\nu}$, $H_0 : \{0,1\}^* \to \{0,1\}^{\iota}$, $H_1 : \{0,1\}^* \to \{0,1\}^{\sigma}$ are cryptographic hash functions and ν, ι, σ are not necessarily different, ι is the size of the secret session key being exchanged. $Mac : \{0,1\}^* \to \{0,1\}^{\nu}$ is a MAC function. The system parameters are publicly known.

During the setup phase the user chooses a password *psw*, the client software generates and securely stores a random salt value z and a random polynomial for the Shamir secret sharing of *psw*. The secret shares s_i, where $i = 1, \ldots, n$ are generated and values $\hat{e}(P, Q_z)$, where $Q_z = H(psw||z)$ and $\hat{e}(s_i P, Q_z)$ are sent and stored by the devices. The password share-based long-lived symmetric secret keys $K_i = H(\hat{e}(s_i P, Q_z))$ are calculated for the authentication during the authentication scheme. If a user wants to set new devices to the smart home system, they need to give the password to the client software, which generates new extra shares s_i for the same polynomial, where $i > n$. This way the construction includes the property of scalability.

3.2 Authentication phase

We assume that $\hat{e}(P, Q_z)$ and $\hat{e}(s_i P, Q_z)$ are stored by the devices and $K_i = H(\hat{e}(s_i P, Q_z))$ are calculated by the client software whenever the password is given. Moreover all IoT devices possess symmetric encryption keys $(\overline{K}_1, \ldots, \overline{K}_n)$ for

authenticated encryption of the messages sent to the manager device. These are short-term keys and exchanged securely in the beginning of the authentication phase.

The authentication phase consists of three main subphases. The first subphase is carried out by the client software. A secret, random authentication value w is chosen and split into shares with Shamir Secret Sharing. These shares and a hash value m_0 based on the w, the password and the salt value are transferred securely to the manager device.

During the second subphase randomly chosen smart home devices calculate their password share-based long-lived symmetric secret keys $K_i = H(\hat{e}(s_i P, Q_z))$, construct and also verify the user's knowledge of the value $\hat{e}(P,Q_z)^{w+psw}$ that is based on the password, the salt and the secret, random authentication value w.

In the third subphase a secret symmetric key is exchanged between the user and the manager device and the user verifies whether the smart home system is able to calculate $\hat{e}(P,Q_z)^{w+psw}$, *i.e.,* whether the devices possess the password shares and the salt.

3.2.1 First subphase

J_v denotes the manager device, which manages the authentication process on the IoT side. It communicates with the user and the other $n-1$ devices. After entering the correct password to the client software, it calculates $K_i = H(\hat{e}(s_i P, Q_z))$, where $\hat{e}(s_i P, Q_z)$ is stored securely and s_i, $i = 1,\ldots,n$ are the secret password shares.

For the Shamir Secret Sharing of a randomly chosen value w, the client software generates a random polynomial $g(x)$ with a degree large enough, where $w = g(0)$. The degree, hence also the threshold depend on the number of devices. The larger the number of devices the larger the threshold is. The threshold - the authentication threshold - is greater than or equal to the password threshold. The secret shares are calculated and denoted by w_i, where $i = 1,\ldots,n$.

I creates the first message M_1 and it is sent to the manager device J_v. I computes $m_0 = H_0(\hat{e}(P,Q_z)^{psw+w})$, where $Q_z = H(psw||z)$ and also calculates $m_v = Mac_{K_v}(r_v \oplus xG \oplus J_v) \oplus w_v||r_v||xG$, where MAC is calculated with the password share-based long-lived symmetric key K_v. Moreover, m_v comprises xG, which is an elliptic curve point represented by a bitstring that is necessary for the key agreement and it is the client message of the elliptic curve Diffie-Hellman key exchange and an r_v random value. The first message also contains $m_i = Mac_{K_i}(r_i \oplus J_i) \oplus w_i||r_i$, where r_i is random and $i = 1,\ldots,n$ and $i \neq v$.

Since w is chosen randomly, for each authentication value w and the shares w_i are different. Therefore values m_i, $i = 0,\ldots,n$ are random.

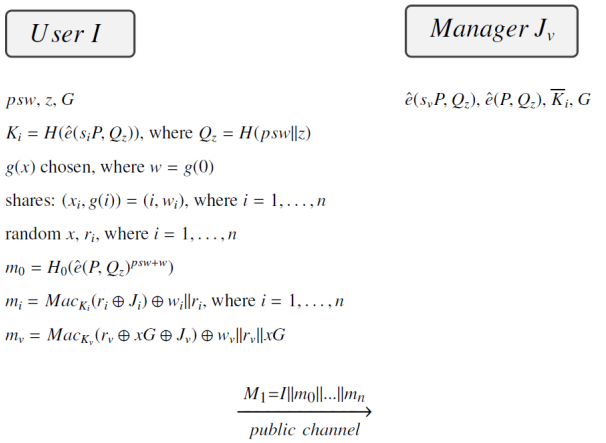

Fig. 2 Authentication - Client process

3.2.2 Second subphase

During the second phase the identity of the user is verified by the smart home system. The device manager chooses m devices randomly, where $k \leq m \leq n$ and k denotes the authentication threshold. These devices - which might include the manager device as well - together authenticate the user.

The device manager also receives m_v. If the manager device is in the set of the chosen devices, then $\hat{m} = m$, *i.e., m* devices verifies value m_0, otherwise $\hat{m} = m+1$, *i.e.,* besides the chosen devices' shares the manager device's share is also applied for the calculation. Hence, the manager device calculates $w'_v = Mac_{K_v}(b \oplus c \oplus J_v) \oplus a$ and $f_v = \hat{e}(P,Q_z)^{w'_v}\hat{e}(s_vP,Q_z)$, where $m_v = a||b||c$, values $\hat{e}(P,Q_z)$, $\hat{e}(s_iP,Q_z)$ are stored by the manager device and K_v is its password share-based long lived key.

After J_v receives message M_1, it forwards each m_i along with I to devices J_{i_j}, where $i_j \in \{1,\ldots,n\}$ and $j \in \{1,\ldots,m\}$. Each device calculates $K_i = H(\hat{e}(s_iP,Q_z))$ and $w'_i = Mac_{K_i}(e \oplus J_i) \oplus d$, where $m_i = d||e$. J_i computes and transmits $f_i = \hat{e}(P,Q_z)^{w'_i}\hat{e}(s_iP,Q_z)$ encrypted with authenticated encryption to the manager device, where values $\hat{e}(s_iP,Q_z)$ and $\hat{e}(P,Q_z)$ are stored by the device. Hence its message authentication and confidentiality are provided by the short-lived symmetric key $\overline{K}_i$. The manager device checks whether $m_0 = H_0(\prod_{l=i_1}^{i_{\hat{m}}} f_i^{t_i})$ holds, where $t_j = \prod_{l=i_1, j\neq l}^{i_{\hat{m}}} \frac{x_l}{x_l - x_j}$ and $\prod_{l=i_1}^{i_{\hat{m}}} f_i^{t_i} = \hat{e}(P,Q_z)^{w+psw}$. If the equality holds the manager device verifies the identity of the user and the integrity of $c = xG$.

3.2.3 Third subphase

Figure 4 demonstrates the steps of the third subphase. J_v generates a random value $y \in \mathbb{Z}_n^*$ and computes the secret session key $ssk = H_0(yxG)$. J_v calculates response $M_2 = h||yG$, where $h = H(ssk||yG||\prod_{i=i_1}^{i_{\hat{m}}} f_i^{t_i})$ and yG is the manager device message in the EC Diffie-Hellman key exchange.

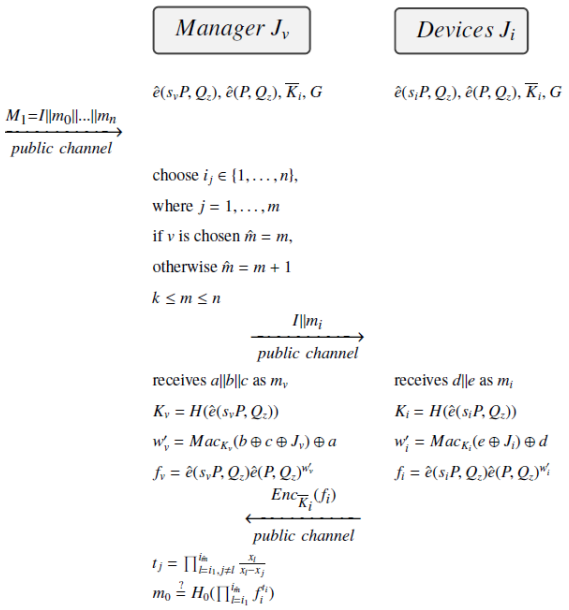

Fig. 3 Authentication - Devices' process

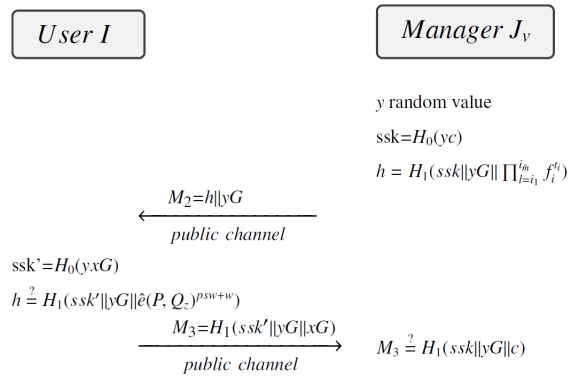

Fig. 4 Authentication - Final process

I receives $M_2 = h||yG$ and calculates the secret session key $ssk' = H_0(yxG)$ and $h' = H(ssk'||yG||\hat{e}(P,Q_z)^{w+psw})$. If $h = h'$, then I confirms that manager device J_v knows the secret session key, and the IoT devices possess the password share-based long-lived secret keys, hence they are authenticated.

Eventually, $M_3 = H_1(ssk'||yG||xG)$ is computed and forwarded to J_v. J_v verifies the message received from the client and if it is correct, J_v confirms that I knows the secret session key.

In the authentication phase, the random value $\hat{e}(P,Q_z)^{w+psw}$ can be calculated by the manager device only if the devices possess the password share-based long-lived symmetric keys, hence the user is able to verify the identity of multiple devices just by checking h. On the other hand, m_0 is checked by calculating $\hat{e}(P,Q_z)^{psw+w}$ involving keys K_i. Correct m_0 proves that the user knows psw and the client software accesses the salt value z, hence the identity of the user is verified as well. Value r_i ensures that the MAC value m_i is fresh for every authentication in order to avoid replay attack. Basically, the secret session key is created via an authenticated key agreement protocol based on random values (x, y) generated by the user and the edge. These values are sent securely so the attacker is not able to gain any information about them.

The authentication phase is also proved to be efficient in terms of time complexity since there are only symmetric encryptions and exponentiations. Both sides use hash, MAC and xor operations, which are also fast operations. Our protocol can be considered robust, because m out of n IoT devices are required for authentication. If a device is not available, the manager device will not choose it.

4 Security analysis

In this section a detailed security analysis of the proposed AKC protocol is provided. One of the indispensable security requirements is the mutual authentication of the participants. Secure mutual authentication of participants prevents adversaries from impersonating a legal user or the device manager and gaining illegal access to sensitive data.

Another security goal is key secrecy, *i.e.,* an adversary should not possess any information about the new key. During a protocol run, a new randomly chosen session key should be exchanged between the participants, a protocol execution could not be successfully completed with an old key exchanged before. At the end, parties should be able to verify that the other party knows and is able to use the new session key. We also consider known-key security and forward secrecy properties. Known-key security preserves the security of session keys after disclosure of a session key. Disclosure of a session key should not jeopardize the security of other session keys. Forward secrecy holds if long-term secrets of one or more entities are compromised and the secrecy of previous session keys is not affected.

Formal methods have been proved to be a good choice for uncovering flaws of incorrectly designed security protocols. There are many tools available that can analyse and identify attacks against protocols, such as Automated Validation of Internet Security Protocols and Applications [3] and ProVerif [14]. In the following subsection we introduce AVISPA, which is an automated formal verification tool used to verify the security properties of the protocols. After presenting AVISPA, we provide the security analysis.

4.1 AVISPA

AVISPA contains four different formal verification approaches (i.e., On-the-fly Model-Checker, Constraint-Logic-based At-

tack Searcher, SAT based Model-Checker and Tree Automata-based Protocol Analyser) that can formally validate security properties of a protocol. It applies the High-Level Protocol Specification Language (HLPSL) which is a modular, role-based language. With the use of HLPSL correct protocol behavior described in the specification is achieved.

The On-the-fly Model-Checker (OFMC) [9–11, 30] executes protocol forgery and bounded session verification as well, where OFMC considers both typed and untyped protocol models. OFMCâĂŹs effectiveness is due to the integration of a number of symbolic, constraint-based techniques, which are correct and complete, in the sense that no attacks are lost nor new ones are introduced by them. These techniques are the lazy intruder technique or the constraint differentiation technique. OFMC also implements a number of efficient search heuristics. It also provides support for modeling an intruder who is capable of performing guessing attacks on weak passwords, and for the specification of algebraic properties of cryptographic operators. OFMC can be employed for protocol falsification and also for verification for a bounded numbers of sessions âĂŤ without bounding the messages an intruder can generate.

The Constraint-Logic-based Attack Searcher (CL-AtSe) is different from the OFMC since it uses constraint solving to perform the protocol falsification and verification for bounded numbers of sessions [23–27]. The protocol messages can be typed or untyped, and the pairing can be considered to be associative or not. It is significant that several properties of the XOR operator can be handled as well. CL-AtSe is built in a modular way and is thus open to extensions for handling algebraic properties of cryptographic operators. CL-AtSe performs several kinds of optimizations to reduce, and often eliminate redundancies or useless branches in the protocol's symbolic execution.

In AVISPA the SAT-based Model-Checker (SATMC) [4–7] could be also applied which considers the typed protocol model and performs both protocol falsification and bounded session verification by reducing the input problem to a sequence of invocations and state-of-the-art SAT solvers. Finally, the TA4SP (Tree Automata based on Automatic Approximations for the Analysis of Security Protocols) backend [15, 16] performs unbounded protocol verification by approximating the intruder knowledge with the help of regular tree languages and rewriting. Genet and Klay introduced an extension of an approximation method based on tree automata [32, 33] for verifying security protocols.

Internally, the attack conditions are specified in terms of temporal logic, but useful and concise macros are provided for the two most frequently used security goals, authentication and secrecy. Among the goal facts *secrecy* declares which values should be kept secret. This goal declaration in the goal section describes that anytime the intruder learns a secret value, even if it is not explicitly given, then it should be considered an attack.

The *witness and request events* are goal facts related to authentication. They are used to check whether a participant is right in believing that its intended peer is present in the current session, has reached a certain state, and agrees on a certain value, which typically is fresh. They always appear in pairs with identical third parameter.

4.2 Security validation

In this part we analyse the security properties of our protocol. Our main goal besides mutual authentication of participants is providing that at the end of the protocol the attacker is not able to gain any information about the exchanged new session key. We formalize the protocol, the security goals and the full version of them can be checked in the appendix. The following part shows the user's role, the user's steps in the protocol formalized in AVISPA. We apply the OFMC and CL-AtSe and execute the attacker simulation. The results of the security analysis show that the attacker cannot impersonate the legal participants or get the session key.

init
State:=0
transition
1.State=0 /\ RCV(start) =|>
State':=2 /\ R1':=new() /\ R2':=new() /\ X':=new() /
W1':=new() /\W2':=new()/
Mac1':= H(Kac.R1') / Mac2':=H(Kab.R2'.exp(G,X'))/
W':=exp(exp(exp(P,Psw),W1'),W2')/
M1':=H(W')/
SND(xor(Mac2',W2').R2'.exp(G,X').H(W')) /\ SND(xor(Mac1',W1').R1')
/
witness(U,DM,user_dm_w,W')
5.State=2 /\ RCV(exp(G,Y')) =|>
State':=7 /\ SSK':=H(exp(exp(G,Y'),X))
6.State=7 /\ RCV(HE') /\ H(W.SSK.exp(G,Y))=HE' =|>
State':=9 /\ Secret':=new() /\ M3':=H(SSK.exp(G,Y).exp(G,X)) /
secret(Secret', sec:1, {U,DM}) /\ SND(M3'.{Secret'}_SSK) /
request(U,DM,user_dm_SSK,SSK)

Let us move on to the formalization of the key secrecy and authentication goals with the AVISPA tool. We formalize the protocol in HLPSL. This language is based on roles: basic roles for representing each participant role, and composition roles for representing scenarios of basic roles. Each role is independent from the others, getting some initial information by parameters, communicating with the other roles by channels. The intruder is modeled using the DolevâĂŞYao model. In AVISPA we can apply the secret, witness and request goal facts. We use these facts to demonstrate that our protocol is secure and we verify the m_0, h, SSK values in the

AVISPA model . The secret is used to show that the session key (SSK) is secret and witness and request serve to prove authentication of participants (m_0, h). In the goal section four goals for authentication and one goal for secrecy are specified:

- secrecy_of sec_1
- authentication_on user_dm_w
- authentication_on dm_user_He
- authentication_on dm_user_SSK
- authentication_on user_dm_SSK

The m_0 value contains the password and w fresh value and we try to find an attack with the authentication_on user_dm_w. The authentication_on dm_user_He goal is similar to authentication_on user_dm_w and we would like to check He (which we denote with h in the protocol) value. Finally, the authentication_on dm_user_SSK and authentication_on user_dm_SSK for the verification of the key confirmation requirement, therefore the mutual authentication is achieved in our protocol as well. In our protocol we have used witness and request for three purposes:

- The user authenticates the device manager on the value of He
- Device manager authenticates the user on the value of w
- The device manager authenticates the user and vica versa on the value of SSK. We abuse strong authentication on SSK here to express that SSK should be generated freshly (and not replayed).

We formalized these statements:

- $secret(Secret, sec\_1, \{U, DM\})$
- $witness(DM, U, dm\_user\_He, HE)$
- $witness(U, DM, user\_dm\_w, W')$
- $request(DM, U, dm\_user\_SSK, SSK)$
- $request(U, DM, user\_dm\_SSK, SSK)$

We describe an example of these statements. The goal fact $secret(Secret, sec\_1, \{U, DM\})$ declares the information "Secret" as secret shared by the agents of the set containing U and DM, moreover $sec1$ protocol id identifies the secrecy goal in the goal section. We check with this fact whether the intruder can calculate the session key SSK. The goal fact $witness(DM, U, dm\_user\_He, HE)$ is for a (weak) authentication property of the device manager DM by user U on HE. The fact declares that the device manager, who is an agent, is a witness to the information HE. The goal will be identified by the constant dm_user_He in the goal section. The goal fact $request(U, DM, user\_dm\_SSK, SSK)$ is for a strong authentication property of the device manager DM by user U on SSK, it declares that agent user U requests a check of the value SSK and this goal will be identified by the constant $user\_dm\_SSK$ in the goal section. The declaration witness represents that value h in the protocol is fresh and generated by the device manager for the user and request represents the user's acceptance of the session key (SSK) that was created by the device manager for the user. The other request and witness statements are working in a similar way. We need to require the strong authentication which prevents replay attacks. Applying the back-ends model checker (OFMC, ATSE) we check whether the intruder can execute the replay attack. Using the DolevâĂŞYao model the back-ends model checkers verify whether there is any man-in-the-middle attack possible by the intruder. If any of the back-end tools finds a trace in which the request event is preceded by a witness event originated by an agent other than U, an attack will be reported.

In OFMC's case, the total number of nodes visited is 2262, while the depth is 4 piles with 5.48 seconds search time. The results of CL-AtSe protocol analysis demonstrate that the number of states analyzed is 26 out of which 15 states can be reached, the translation time is 0.07 seconds and the computation time is 0.15 seconds. The output proves that the proposed protocol is safe against active and passive attacks. The known-key and forward secrecy hold in the proposed protocol as well. The parameters xG and yG of the session key are chosen independently and randomly for every protocol run, hence the new session keys are also independent and random. The session key can be calculated by the adversary from xG and yG only if he or she computes ECCDH function. The adversary faces ECCDH assumption for the previous session keys even if long-term secret keys are compromised.

5 Computation and communication cost analysis

Since efficiency is an important aspect, evaluation of the computation and communication cost of the protocol is essential. Overall, the session key is generated by ECDH key exchange, and there are hash, MAC, xor calculations and exponentiations, which are considered to be fast operations.

In the efficiency analysis the number of devices participating in the identity verification is an important parameter.

5.1 Devices

We implemented the steps of the protocol in python program language in order to test it in a real environment. We applied and focused on three devices (PC, Raspberry PI 4, ESP32). We used a PC with the following parameters: AMD Ryzen 5 2600 - 6 cores, 12 threads, 3.4 GHz - 3.9 GHz 16 GB RAM and an M.2 NVMe SSD with 3200 MB/s writing - 3500 MB/s reading speed. The Raspberry Pi is a popular small on-board computer (SBC). In a smart home, we assume that a manager device is a more powerful device that can be matched with a Raspberry PI. Raspberry has all the software you need for basic computing. Although Raspbian is

the OS officially recommended by the manufacturer, countless other operating systems are available. In our case, we apply a RasPi, which includes Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz processor. It has 2 GB RAM and a class 10 memory card. The ESP32-WROOM-32 is a high-performance, generic Wi-Fi + BT + BLE MCU module that targets a wide range of applications, from low-performance sensor networks to the most demanding tasks such as audio encoding, music streaming and MP3 decoding. The essence of the module is the ESP32-D0WDQ6 chip. The embedded chip is scalable and adaptive. There are two CPU cores that can be controlled separately and the frequency of the CPU clock can be adjusted from 80 MHz to 240 MHz.

5.2 Communication and computation costs of the protocol

In this section, we detail the communication and computation costs of the protocol. During the identity verification, we can parallelize the operations on the IoT side. Even if we apply more devices, the time does not increase. Furthermore, this parameter denoted by m can be adjusted dynamically in our protocol.

5.2.1 Computation cost

The following table summarizes the computational requirements for each device:

Operation	User device	Manager Device	IoT Devices
Exponentation	1	1	1
SHA 256	n+6	4	0
AES 128	0	m-1 or m	1
HMAC	n	1	1
EC scalar mult.	n+3	2	0

Table 2 Number of operations

We demonstrate the execution time of the operations in seconds. The authentication protocol can be executed within 1 second even for a large number of devices. In addition to mutual authentication, this execution time also includes the session key exchange and the key confirmation. This is considered acceptable for the user.

Figure 5 shows the execution time for the client depending on the number of IoT devices.

Table 4 shows the computational performance of the authentication phases with different devices. We have selected a threshold authentication system [40] from Table 1 in Section 1.1 that is similar to our system. Their runtime results are compared to our ones for the different number of devices and thresholds. According to [63] in 2022, on average

Operation	PC	ESP32	Raspberry
Exponentation	0,0002143	0,01582	0,001685
SHA 256	0,0000002	0,00002	0,0000008
AES 128	0,0000033	0,00155	0,0000016
HMAC	0,0000011	0,0082	0,0000059
EC scalar mult.	0,002880	0,2945	0,0205

Table 3 Execution time of protocol's operations (second)

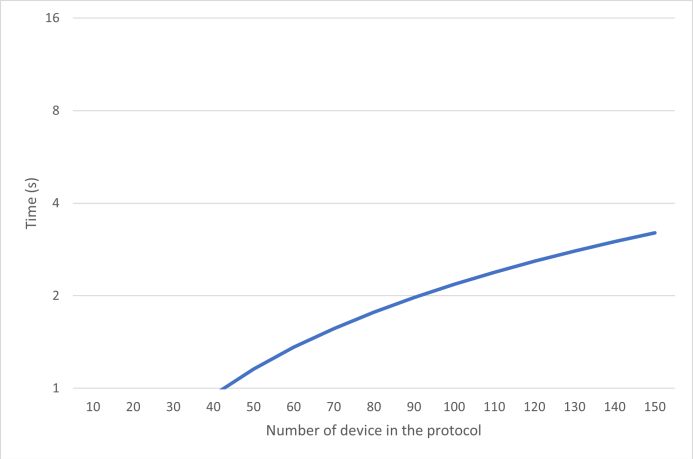

Fig. 5 Execution time for the client with different number of sensors

Threshold	Raspberry+ESP32	PC+ESP32	PC
2-5 Threshold	0,233989	0,0548115	0,0294602
3-6 Threshold	0,2544973	0,0576961	0,0323448
5-10 Threshold	0,3365273	0,0692279	0,0438766

Table 4 Performance evaluation of our Threshold Authentication (in seconds).

500 devices will be connected per household, hence large number of devices and threshold should be considered. Our proposition provides a better result for $n \geq 10$ number of devices and for $m \geq 5$ threshold.

Threshold	2-5	3-6	5-10
Işler, Küpçü - DSPP	0,00806	0,01171	0,01833
Our proposition	0,0150602	0,0150648	0,0150766

Table 5 Performance comparison (in seconds).

Today, achieving computing capacity and adequate security are important considerations for IoT devices. The cost of manufacturing affects the capabilities of these devices, however we need to ensure security. These aspects were also taken into account during the design of our protocol.

5.2.2 Communication cost

We assume that the identifiers of participants are 32-bit. In this case 4,294,967,296 (2^{32}) unique IDs are possible. During authentication, the hash and MAC values are 256 bits, the AES symmetric ciphertexts are 128 bits, and the elliptical curve points are 448 bits. The following table summarizes the bits sent by the client, device manager and the devices. The manager always participates in the authentication. If the manager is among the randomly chosen m devices, then $m-1$ IoT devices are chosen, otherwise m.

Client	$(n+1)\cdot(32+256)+n\cdot 128+448+256$
Manager	(m-1 or m) $\cdot(32+256+128)+256+448$
IoT	(m-1 or m)$\cdot$128

Table 6 communication cost of the protocol

We can see that the costs of communication and computation depend largely on the number of devices involved in the authentication. The computation cost is negligible because the manager device and the client device can execute these operations extremely quickly. Moreover, we emphasize that the IoT devices compute in parallel and this phase does not increase significantly the execution time of the protocol even if we add more devices to the smart home. Figure 6 demonstrates how communication cost and data traffic are growing on the network.

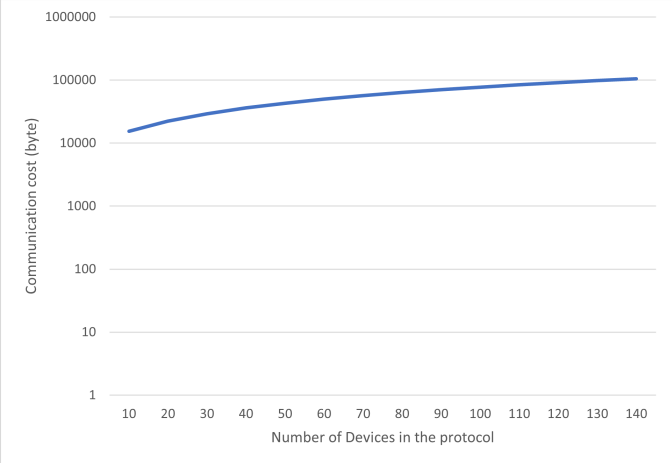

Fig. 6 Communication cost of the protocol with different number of sensors

In order to make communication secure and message size acceptable, the appropriate value has to be specified. We propose to apply around ten to fifty devices for the protocol, because the communication cost is still small and also ensures security.

6 Conclusion

We have introduced a mutual authentication protocol for smart homes. This environment has special characteristics, it contains a device manager and several resource-constrained devices. Most cases the device manager behaves as an edge preprocessing data received from the devices. The number of IoT devices is increasing and some of them might also break down. The proposed protocol is designed for this environment, it is a threshold and password-based distributed AKC protocol that is scalable and robust with a password-only setting. We formalized the protocol in AVISPA and we examined the security of this based on different security goals. The AVISPA did not find any kind of vulnerability or attack hence the protocol achieves the security goals (mutual authentication, key secrecy). Finally, we implemented the protocol and measured the performance of our proposed solution instance.

Compliance with Ethical Standards

Funding:
The presented research has been supported by the SETIT Project (no. 2018-1.2.1-NKP-2018-00004), which has been implemented with the support provided by the National Research, Development and Innovation Fund of Hungary, financed under the 2018-1.2.1-NKP funding scheme. Norbert OlÃąh supported by the ÃŽNKP-21-4 New National Excellence Program of the Ministry for Innovation and Technology from the source of the National Research, Development and Innovation Fund.

Conflict of Interest:
Andrea PintÃl'r-Huszti, Norbert OlÃąh and Szabolcs KovÃącs declare that they have no conflict of interest.

Ethical approval:
This article does not contain any studies with human participants or animals performed by any of the authors.

References

1. Acar, T., Belenkiy, M., Küpcü, A. *Single password authentication.* Comput. Netw. 57(13), 2597âĂŞ2614 (2013)
2. Karim Arabi *Mobile Computing Opportunities, Challenges and Technology Drivers,* IEEE DAC 2014 Keynote, http://www2.dac.com/events/videoarchive.aspx?confid=170&filter= keynote&id=170-103–0&#video
3. A. Armando, Basin D, Boichut Y, Chevalier Y, Compagna L, CuÃl'llar J, Drielsma PH, HÃl'am PC, Kouchnarenko O, Mantovani J, et. al. *The avispa*

tool for the automated validation of internet security protocols and applications. International conference on computer aided verification (Springer), , pp 281âĂŞ285. (2005)

4. A. Armando and L. Compagna. *Automatic SAT-Compilation of Protocol Insecurity Problems via Reduction to Planning.* In Proc. FORTE 2002, LNCS 2529, pp. 210âĂŞ225. Springer, (2002)
5. A. Armando and L. Compagna. *Abstraction-driven SAT-based Analysis of Security Protocols.* In Proc. SATâĂŹ03, LNCS 2919, pp. 257âĂŞ271. Springer, (2003)
6. A. Armando and L. Compagna. *SATMC: a SAT-based Model Checker for Security Protocols.* In Proc. JELIAâĂŹ04, LNAI 3229, pp. 617âĂŞ627. Springer, (2004)
7. A. Armando and L. Compagna. *An Optimized Intruder Model for SAT-based Model-Checking of Security Protocols.* In Proc. ARSPAâĂŹ04. Electronic Notes in Theoretical Computer Science 125(1):91âĂŞ108, (2005)
8. Bagherzandi, A., Jarecki, S., Saxena, N., Lu, Y.: *Password-protected secret sharing.* In: ACM Conference on Computer and Communications Security (2011)
9. D. Basin, S. MÃűdersheim, and L. ViganÃş. *Constraint Differentiation: A New Reduction Technique for Constraint-Based Analysis of Security Protocols.* In Proc. CCSâĂŹ03, pp. 335âĂŞ344. ACM Press, (2003)
10. D. Basin, S. MÃűdersheim, and L. ViganÃş. *Algebraic intruder deductions.* In Proc. LPARâĂŹ05, LNAI 3835, pp. 549âĂŞ564. Springer, (2005)
11. D. Basin, S. MÃűdersheim, and L. ViganÃş. *OFMC: A symbolic model checker for security protocols.* International Journal of Information Security, 4(3):181âĂŞ208, (2005)
12. Steven M. Bellovin and Michael Merritt, *Encrypted key exchange: Password-based protocols secure against dictionary attacks,* In Proceedings of the 1992 IEEE Computer Society Conference on Research in Security and Privacy. IEEE,1992.
13. Bitdefender Whitepaper, *Severe Vulnerability in iBaby Monitor M6S Camera Leads to Remote Access to Video Storage Bucket* https://www.bitdefender.com /files/News/CaseStudies/study/315/Bitdefender-Whitepaper-Severe-Vulnerability-in-iBaby-Monitor-M6S-Camera-interactive.pdf 2019
14. B. Blanchet. *An efficient cryptographic protocol verifier based on Prolog rules.* In 14th IEEE Computer Security Foundations Workshop (CSFW-14), pages 82âĂŞ96, (2001)
15. Y. Boichut, P.-C. HÃľam, and O. Kouchnarenko. *Automatic verification of security protocols using approximations.* Technical Report RR-5727, INRIA, (2005)
16. Y. Boichut, P.-C. Heam, O. Kouchnarenko, and F. Oehl. *Improvements on the Genet and Klay Technique to Automatically Verify Security Protocols.* In Proc. International Workshop on Automated Verification of Infinite-State Systems (AVISâĂŹ2004), pp. 1âĂŞ11, (2004)
17. V. Boyko, P. MacKenzie, and S. Patel. *Provably secure password-authenticated key exchange using diffie-hellman.* In EUROCRYPT 2000. Springer, (2000)
18. Boyen, X. *Hidden credential retrieval from a reusable password.* In: Proceedings of the 4th International Symposium on Information, pp. 228âĂŞ238. ACM (2009)
19. Dan Boneh and Matthew Franklin, *Identity based encryption from the Weil pairing*, Advances in Cryptography - Proceedings of Crypto 2001, Vol 2139 of LNCS, 213–229., (2001)
20. J. Brainard, Ari Juels, Burt Kaliski, and Michael Szydlo, *A New Two-Server Approach for Authentication with Short Secrets*, Proceeding SSYM'03, Proceedings of the 12th conference on USENIX Security Symposium - Volume 12, pp 1-14, 2003.
21. J. Cao, L. Xu, R. Abdallah, and W. Shi, *EdgeOS_H: A home operating system for Internet of everything* In Proc. 37th Int. Conf. Distrib. Comput. Syst., Jun. 2017, pp. 1756âĂŞ1764.
22. N. Chen, R. Jiang, *Security Analysis and Improvement of User Authentication Framework for Cloud Computing*, Journal of Networks, 9(1), pp. 198-203, 2014.
23. Y. Chevalier, R. KÃijsters, M. Rusinowitch, and M. Turuani. *Deciding the Security of Protocols with Commuting Public Key Encryption.* In Proc. ARSPAâĂŹ04. Electronic Notes in Theoretical Computer Science 125(1):55âĂŞ66, (2005)
24. Y. Chevalier, R. KÃijsters, M. Rusinowitch, M. Turuani, and L. Vigneron. *Deciding the Security of Protocols with Diffie-Hellman Exponentiation and Products in Exponents.* In Proc. FST TCSâĂŹ2003, LNCS 2914, pp. 124âĂŞ135. Springer, (2003)
25. Y. Chevalier, R. KÃijsters, M. Rusinowitch, M. Turuani, and L. Vigneron. *Extending the Dolev-Yao Intruder for Analyzing an Unbounded Number of Sessions.* In Proc. CSLâĂŹ2003, LNCS 2803,pp. 128âĂŞ141. Springer, (2003)
26. Y. Chevalier, R. KÃijsters, M. Rusinowitch, and M. Turuani. *An NP Decision Procedure for Protocol Insecurity with XOR.* Theoretical Computer Science 338(1âĂŞ3):247âĂŞ274, (2005)
27. Y. Chevalier and M. Rusinowitch. *Combining Intruder Theories.* In Proc. ICALP 2005, LNCS 3580, pp. 639âĂŞ651. Springer, (2005)
28. A. J. Choudhury, P. Kumar, M. Sain, *A Strong User Authentication Framework for Cloud Computing*, Proceedings of IEEE Asia -Pacific Services Computing Conference, pp. 110-115, (2011)

29. Stebila, Douglas J.; UDUPI, Poornaprajna V.; SHANTZ, Sheueling Chang, *Multi-factor password-authenticated key exchange.* U.S. Patent No 8,776,176, 2014.
30. P. Hankes Drielsma and S. MÃűdersheim. *The ASW Protocol Revisited: A Unified View.* In Proc. ARSPAâĂŹ04. Electronic Notes in Theoretical Computer Science 125(1):141âĂŞ156, 2005.
31. Milan Fránik, Milos Cermák, *Serious flaws found in multiple smart home hubs: Is your device among them?* https://www.welivesecurity.com/2020/04/22/serious-flaws-smart-home-hubs-is-your-device-among-them/, 2020
32. T. Genet and F. Klay. *Rewriting for cryptographic protocol verification.* In Proc. CADEâĂŹ00, LNCS 1831, pp. 271âĂŞ290. Springer, 2000.
33. T. Genet and V. Viet Triem Tong. *Reachability Analysis of Term Rewriting Systems with Timbuk.* In Proc. LPARâĂŹ01, LNCS 2250, 2001.
34. Hoang, V. H., Lehtihet, E., & Ghamri-Doudane, Y. *Password-based authenticated key exchange based on signcryption for the Internet of Things.* In 2019 Wireless Days (WD) (pp. 1-8). IEEE.
35. Jiang, Y., Huang, Q., Zhu, C., Wang, Y., & Shen, J. *An Efficient Key Agreement Protocol for Secure Group Communications Using Periodic Array.* In 2019 IEEE 13th International Conference on Anti-counterfeiting, Security, and Identification (ASID) (pp. 36-40). IEEE.
36. A. Huszti, N. Oláh, *A simple authentication scheme for clouds*, Proceedings of IEEE Conference on Communications and Network Security (CNS), (2016), Pages: 565 - 569.
37. M. S. Hwang, L. H. Li, *A new remote user authentication scheme using smart cards*, IEEE Transactions on Consumer Electronics, 46(1), pp. 28-30, 2000.
38. Işler, D., & Küpçü, A. *Threshold single password authentication.* ESORICS Data Privacy Management, Cryptocurrencies and Blockchain Technology, pp. 143-162, 2017
39. Ibrahim, M. H. *Octopus: An Edge-fog Mutual Authentication Scheme,* IJ Network Security, 18(6), 1089-1101. (2016).
40. Işler, D., & Küpçü, A. *Distributed Single Password Protocol Framework.* IACR Cryptol. ePrint Arch., 2018, 976.
41. Iqbal, W., Abbas, H., Rauf, B., Abbas, Y., Amjad, F., & Hemani, A. (2021). *PCSS: Privacy Preserving Communication Scheme for SDN Enabled Smart Homes.* IEEE Sensors Journal.
42. Jarecki, S., Kiayias, A., & Krawczyk, H. *Round-optimal password-protected secret sharing and T-PAKE in the password-only model*, In International Conference on the Theory and Application of Cryptology and Information Security (pp. 233-253). Springer, Berlin, Heidelberg. 2014.
43. Katz, J., Ostrovsky, R., Yung, M. *Efficient password-authenticated key exchange using human-memorable passwords.* In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 475âĂŞ494. Springer, Heidelberg (2001). https://doi.org/10. 1007/3-540-44987-6 29
44. Khan, M. A., & Salah, K. *IoT security: Review, blockchain solutions, and open challenges.* Future Generation Computer Systems, 82, pp 395-411., 2018.
45. Kolias, C., Kambourakis, G., Stavrou, A., & Voas, J. *DDoS in the IoT: Mirai and other botnets.* Computer, 50(7), pp 80-84. , 2017.
46. W. C. Ku, S. M. Chen, *Weaknesses and improvements of an efficient password based remote user authentication scheme using smart cards,* IEEE Transactions on Consumer Electronics, 50(1), pp. 204-207, 2004.
47. LHN: Zipato Smart Home Hub Exploits Discovered That Could Allow An Attacker to Open iOT Door Locks *https://latesthackingnews.com/2019/07/05/hackers-can-exploit-zipato-smart-home-hub-flaws-to-break-in/, 2019*
48. Li, W., Cheng, H., Wang, P., & Liang, K. (2021). *Practical Threshold Multi-Factor Authentication.* IEEE Transactions on Information Forensics and Security.
49. L. Lin, X. Liao, H. Jin and P. Li, *Computation Offloading Toward Edge Computing,* In Proceedings of the IEEE, vol. 107, no. 8, pp. 1584-1607, Aug. 2019, doi: 10.1109/JPROC.2019.2922285.
50. Z. Ling, J. Luo, Y. Xu, C. Gao, K. Wu and X. Fu, *Security Vulnerabilities of Internet of Things: A Case Study of the Smart Plug System,* In IEEE Internet of Things Journal, vol. 4, no. 6, pp. 1899-1909, Dec. 2017, doi: 10.1109/JIOT.2017.2707465.
51. P. MacKenzie, T. Shrimpton, and M. Jakobsson. *Threshold password-authenticated key exchange.* In CRYPTO 2002. Springer, 2002.
52. S. Marksteiner, V. J. Exposito Jimenez, H. Vallant, and H. Zeiner. *An Overview of Wireless IoT Protocol Security in the Smart Home Domain*, In: 2017 Joint 13th CTTE and 10th CMI Conference on Internet of Things Business Models, Users, and Networks, Copenhagen, 2017, pp. 1-8. doi: 10.1109/CTTE.2017.8260940. http://ieeexplore.ieee.org/document/8260940/
53. Menezes, A. J., Okamoto, T., & Vanstone, S. A. (1993). *Reducing elliptic curve logarithms to logarithms in a finite field.* iEEE Transactions on information Theory, 39(5), 1639-1646.
54. Ralph C. Merkle, *A Digital Signature Based on a Conventional Encryption Function*, Advances in Cryptology - CRYPTO '87, Lecture Notes in Computer Science, **293**, (1987), pp. 369-378.

55. Young Man Park and Sang Gyu Park, *Two factor authenticated key exchange (TAKE) protocol in public wireless LANs,* IEICE Transactions on Communications, E87-B(5):1382âĂŞ1385, May 2004.
56. David Pointcheval and SÃl'bastien Zimmer, *Multi-factor authenticated key exchange,* In Steven M. Bellovin and Rosario Gennaro, editors, Applied Cryptography and Network Security (ACNS) 2008, LNCS, volume 5037, pp. 277âĂŞ295. Springer, 2008.
57. Di Raimondo, Mario, and Rosario Gennaro. *Provably secure threshold password-authenticated key exchange*, International Conference on the Theory and Applications of Cryptographic Techniques. Springer, Berlin, Heidelberg, 2003.
58. Rathore, M. Mazhar; BENTAFAT, Elmahdi; BAKIRAS, Spiridon. *Smart Home Security: A Distributed Identity-Based Security Protocol for Authentication and Key Exchange.* In: 2019 28th International Conference on Computer Communication and Networks (ICCCN). IEEE, pp. 1-9., 2019.
59. Saeed, M.E.S., Liu, QY., Tian, G. et. al. *AKAIoTs: authenticated key agreement for Internet of Things.* Wireless Netw 25, 3081âĂŞ3101 (2019). https://doi.org/10.1007/s11276-018-1704-5
60. Samsung SmartCam. *https://www.exploitee.rs/index.php/%20Samsung%20SmartCam%E2%80%8B#Fixing%20Password%20Reset%20.22Pre-Auth.22, August 2014.*
61. Shen, J., Chang, S., Shen, J., Liu, Q., & Sun, X. , *A lightweight multi-layer authentication protocol for wireless body area networks,* Future Generation Computer Systems, 78, 956-963., (2018)
62. Alladi, Tejasvi, et. al. *Consumer IoT: Security vulnerability case studies and solutions.* IEEE Consumer Electronics Magazine, 9.2: 17-25., 2020.
63. GU, Tianbo; MOHAPATRA, Prasant. *Bf-iot: Securing the iot networks via fingerprinting-based device authentication,* In: 2018 IEEE 15Th international conference on mobile ad hoc and sensor systems (MASS). IEEE, p. 254-262., 2018.
64. R. Trimananda, A. Younis, B. Wang, B. Xu, B. Demsky and G. Xu, *Vigilia: Securing Smart Home Edge Computing,* 2018 IEEE/ACM Symposium on Edge Computing (SEC), Seattle, WA, 2018, pp. 74-89, doi: 10.1109/SEC.2018.00013.
65. Lee, Yunho, Seungjoo Kim, Dongho Won Enhancement of two-factor authenticated key exchange protocols in public wireless LANs. Computers & electrical engineering 36.1 (2010): 213-223.
66. Vécsi, Á., Bagossy, A., & Pethö, A., *Cross-platform Identity-based Cryptography using WebAssembly,* Infocommunications, 31., (2019).
67. G. Wassermann. *ZyXEL NBG-418N, PMG5318-B20A and P-660HW-T1 routers contain multiple vulnerabilities.* http://www.kb.cert.org/vuls/id/870744, October 2015.
68. Wu, T., Lee, Z., Obaidat, M.S., Kumari, S., Kumar, S., Chen, C. *An authenticated key exchange protocol for multi-server architecture in 5G networks.* IEEE Access 8, 28096âĂŞ28108 (2020). https://doi.org/10.1109/ACCESS.2020.2969986
69. I. Zavalyshyn, N. O. Duarte, and N. Santos, *HomePad: A privacy-aware smart hub for home environments* In Proc. IEEE/ACM Symp. Edge Comput., Oct. 2018, pp. 58âĂŞ73

Appendix

- The elliptic curve what we use: $y^2 = x^3 + x$, which is a supersingular curve
- The prime : p = 731329576256467855322039941411215536384068289627312830254310277821058411810144462486413246228592183502383911176278505421042514024101864935444571849103 97
- The order: q = 730750818665451459101842416358141509827966402561

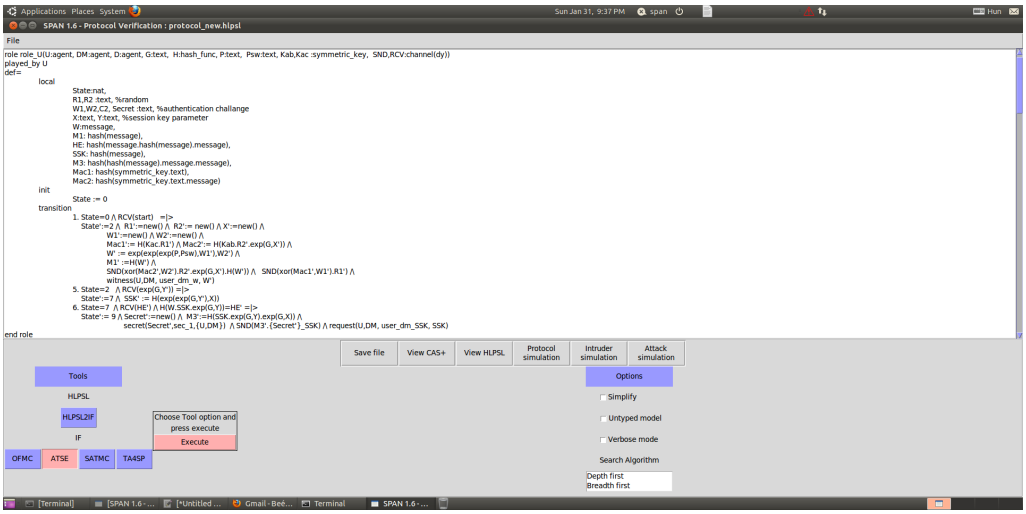

Fig. 7 HLPSL specification of user's role

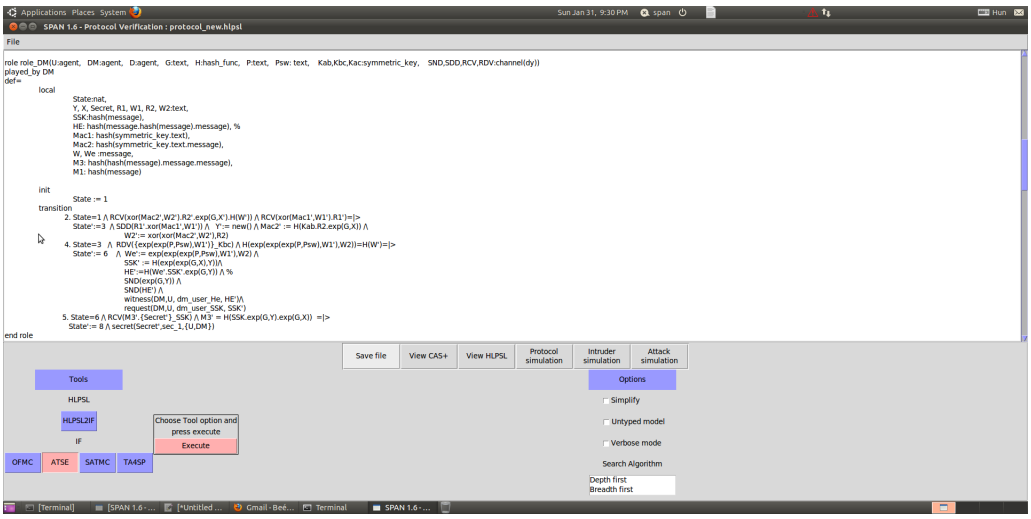

Fig. 8 HLPSL specification of device manager's role

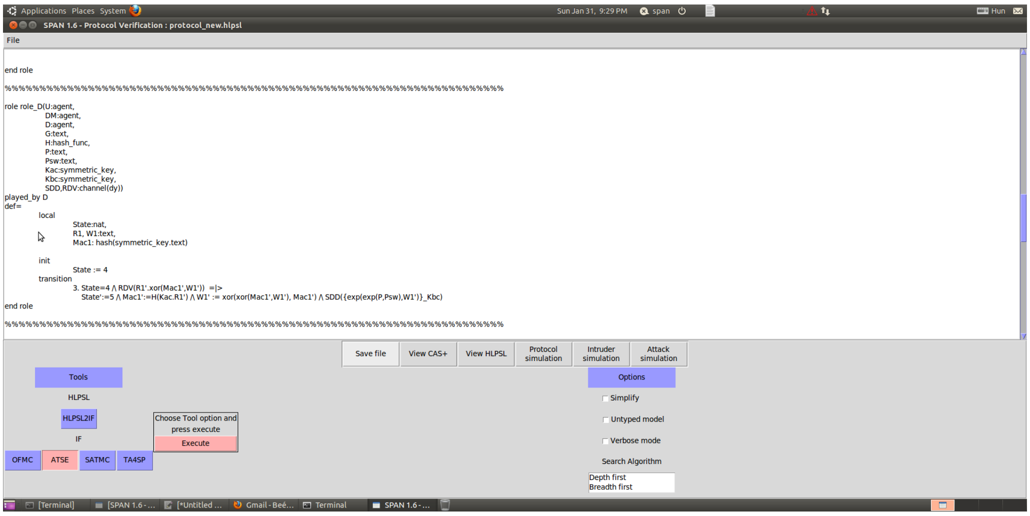

Fig. 9 HLPSL specification of device's role

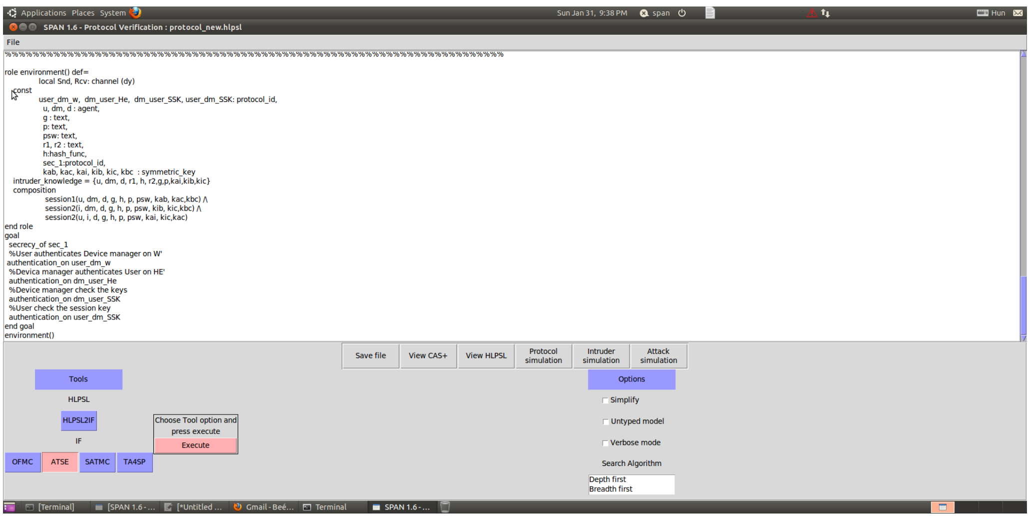

Fig. 10 HLPSL Specification of role environment

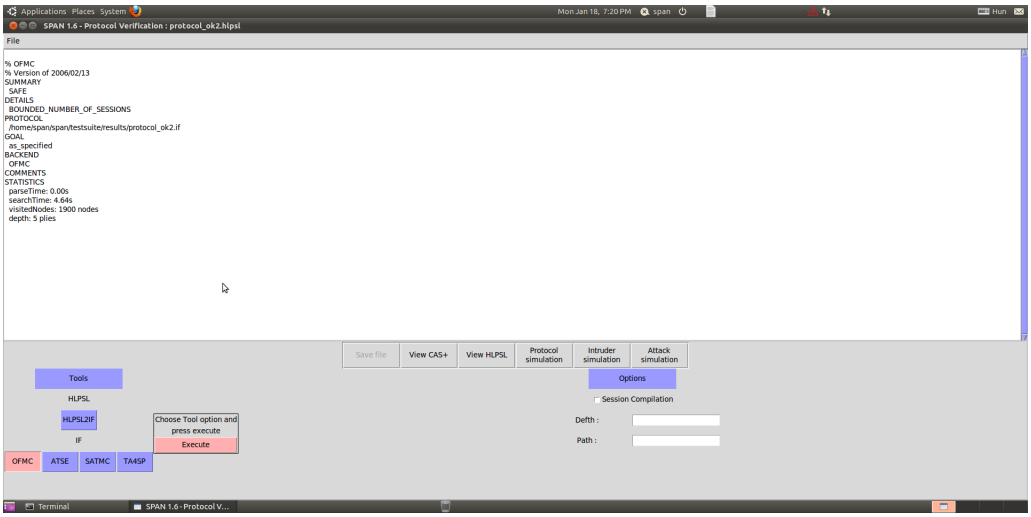

Fig. 11 OFMC output of the proposed protocol verified in AVISPA

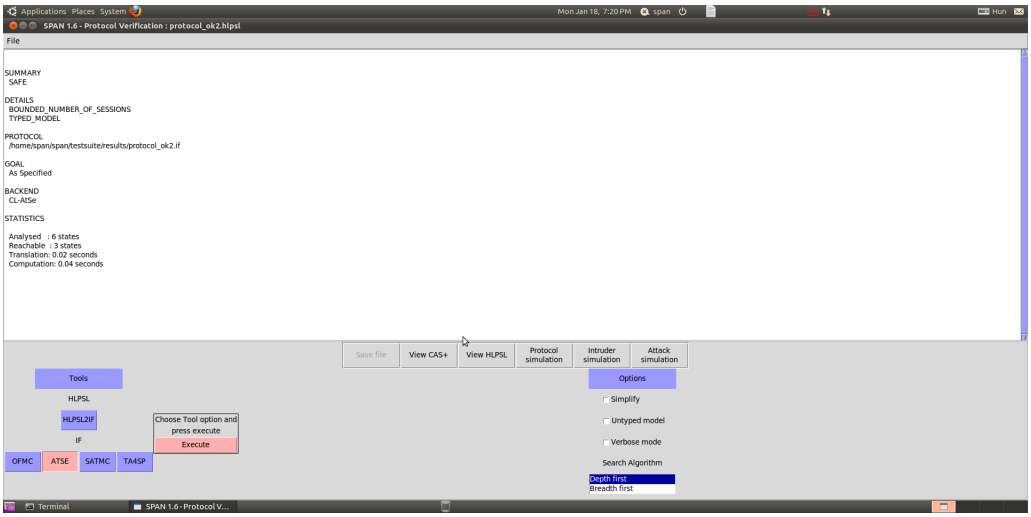

Fig. 12 CL-AtSe output of the proposed protocol verified in AVISPA