Homomorphic Hash and Blockchain Based Authentication Key Exchange Protocol for Strangers

Hailong Yao
School of Electronic and Information Engineering
Lanzhou City University
Lanzhou, Cina
Hailong.Yao@outlook.com

Caifen Wang
College of Computer Science & Engineering
Northwest Normal University
Lanzhou, Cina
wangcf@nwnu.edu.cn

Bo Hai
School of Electronic and Information Engineering
Lanzhou City University
Lanzhou, Cina
Haibo@lzcu.edu.cn

Shiqiang Zhu
School of Electronic and Information Engineering
Lanzhou City University
Lanzhou, Cina
Zhusq@lzcu.edu.cn

***Abstract*—Modern communication technologies and cryptographic technologies have pushed social networks into the virtual world, but they have also ensured the real existence of social proximity. In the evaluation of social proximity, some decentralized scenarios require the participants bootstrapping tust, and the existing authentication key exchange scheme can hardly satisfy the above requirements. In this study, we have designed a homomorphic hash and Blockchain based authenticated key exchange protocol with privacy protection, and prove its security under the standard model based on hash one-way, discrete logarithm and Blockchain transaction-level security assumption, and discussed the attack that the proposed scheme can resist. Compared with the existing scheme, the proposed scheme does not need the default Unit of Trust, is safer and more flexible, suitable for the scenarios that require strangers to bootstrap trust.**

Keywords—Authenticated Key Exchange Protocol, Bootstrap Trust, Privacy Protection, Homomorphic Hash

I. Introduction

Everything benefits from the sharing of information, things are known because of it, and species continue because of it. The birth and inheritance of human civilization is inseparable from the sharing of information. In modern society, people are spreading all sorts of information, either intentionally or unintentionally. Such as short tweet, video and location information and so on. Although the dissemination of information is personal freedom, some information is inconsistent with the law, and some of the information is detrimental to personal privacy. Some information can be broadcast, some information needs to be exchanged in person or in close proximity, and some information needs to be exchanged anonymously. Therefore, it is a worthwhile issue to transmit specific information to specific recipients in a specific manner [1]. In some scenarios where anonymous and secure data exchange between strangers is required, it is ideal for participants to run a Authentication Key Exchange (AKE) protocol with bootstrapping trust.

AKE protocol can resist Man-in-the-middle Attacks, Replay Attacks and Impersonation Attacks since mutual authentication between participants. AKE has become very popular in the Internet in the last few years. Unfortunately, most of the existing AKE based on some Root of Trust or Chain of Trust between the involved parties, such as prior knowledge on a cryptographic key or password [2]-[7] or Trusted Third Parties (TTP) [8][9]. In a P2P-like scenarios, the above Unit of Trust (UoT) cannot be deployed. To address these issues, many decentralized solutions are proposed[10]-[12]. These schemes use key directories that store bindings between identities and public keys to achieve consistency. While these systems provide a reasonable user experience and do not depend on any TTP, restricting parties can only use the registered keys and thus complicating key management. In particular, these schemes do not apply to scenarios where strangers need to bootstrap trust. Dong et al. designed a secure social proximity computation protocol [13], which can identify potential friends on the premise of ensuring privacy, but it requires TTP initialization parameters before running. McCorry et al. proposed two protocols on the basis of ECDH [14] and YAK [15], for secure communication between Bitcoin users in a post-transaction scenario without requiring any TTP or additional authentication credentials [16]. Bui et al. present a family of key exchange protocols that utilizes the global consistency property of the public ledgers [17]. These protocols require secure out-of-band channel for sharing parameters, when public identity are not available.

In this paper, we present a homomorphic hash and Blockchain based AKE protocol (HBAKE), which can meet the security needs and the computational demands of scenarios that require strangers to bootstrap trust in an untrustworthy environment. Multiple unfamiliar parties authenticate common secret holders through Blockchain transactions and complete the exchange of session keys on the promise of privacy protection. The authentication here means identity authentication of participants who hold the common secrets and transaction-level certification of the key agreement process. In this case, users who need to bootstrap trust use the common interest event as a carrier to publish the hash value of the common secret along with the tags corresponding to the secret in the form of a Blockchain transaction. Any user with similar needs can guess tags for the keyword initializing the Bloom filter and searching for common interest events on the Blockchain, if it is found that the holder of the common interest event uses the homomorphic hash function value of the common secret to construct the transaction leading to the potential partner to initiate the common secret secure matching, which ultimately calculates session secrets based on common strings.

The rest of the paper is organized as follows. Section II introduces the basics needed to understand this paper. We describe the detail of our proposed scheme in Section III and prove its security under the Standard Model. Section IV shows

978-1-5386-8034-6/18/$31.00 ©2018 IEEE
DOI 10.1109/CBD.2018.00051

some applications of the proposed scheme. The paper is concluded in Section V.

II. Preliminaries

In this section, we review some preliminary knowledge used in this paper, including the homomorphic hash functions, Blockchain and other cryptographic primitives.

A. Homomorphic Hashing Function

A homomorphic hash function hh: $F_n \rightarrow G_q$ is defined as a collision-resistant hash function satisfying the homomorphism in addition to the properties of a universal hash function uh: $(0, 1)^* \rightarrow (0, 1)^l$. Where, F is finite field and G_q is multiplicative group of prime order q.

1) One-way: It is computationally hard to find $hh^{-1}(\boldsymbol{x})$.

2) Collision free: It is computationally hard to find vectors $\boldsymbol{x}, \boldsymbol{y} \in F^n$ $(x \neq y)$, such that $hh(x) = hh(y)$.

3) Homomorphism: For any $\boldsymbol{x}$, $\boldsymbol{y} \in F^n$, $hh(\boldsymbol{x} \circ \boldsymbol{y}) = hh(\boldsymbol{x}) \circ hh(\boldsymbol{y})$, "∘" is either an "+" or an "·".

Krohn et al. [18] construct a homomorphic hash function satisfying additive homomorphism. Let n elements $g_1, g_2, \ldots, g_m$ be selected randomly from G_q. Then, for any vector $\boldsymbol{x}(x_1,\ldots,x_n) \in Z_q^n$, the homomorphic hash is defined as $hh(\boldsymbol{x}) = \prod_{i=1}^{n} g_i^{x_i}$.

For any vector $\boldsymbol{x}$, $\boldsymbol{y} \in Z_q^n$, $hh(\boldsymbol{x}+\boldsymbol{y}) = \prod_{i=1}^{n} g_i^{x_i+y_i} = \prod_{i=1}^{n} g_i^{x_i} \cdot \prod_{i=1}^{n} g_i^{x_i} = hh(\boldsymbol{x}) \cdot hh(\boldsymbol{y})$, and the collision-resistance property is derived from the discrete logarithm assumption over G_q.

B. Blockchain Virtual Interface and Transaction-level Security

Blockchain is a new type of distributed ledger technology that derives from Bitcoin [19] underlying technology. The open distributed ledger running on the P2P network is maintained by all ledger members, each node can copy a duplicate of the complete record. The nodes participating in the system may not belong to the same organization and do not need to trust each other. If we purposely ignore the implementation details of the Blockchain and abstract it into a virtual interface that provides data integrity, non-repudiation, and user privacy guarantees. We call the mechanism of transaction integrity, non-falsification, and user privacy protection provided by the blockchain virtual interface as transaction-level security.

C. Common Interest Event

In social networks, strangers with common interests are more likely to trust and connect with each other. Our scheme need a tool that helps us manage the information used to generate shared secrets on the Blockchain. The information must have well-known names for easy searching, but their content is not necessarily the same, they are named Common Interest Events (CIE). The CIE is used to share secrets between entities in untrustworthy environments which consists of a Tag (T) and Common String (CS). CS is the parameter used to generate the shared secret. CIE is defined as follows:

Definition 1 the set of **CIE** for any of the entities A and B in the network is denoted:

CIE = { CIE_k | CIE_k $(\mathrm{T}_k, \mathrm{CS}_k)$ = $(\mathrm{CIE}_{\mathrm{A}i}$ $(\mathrm{T}_{\mathrm{A}i}, \mathrm{CS}_{\mathrm{A}i}) \cap \mathrm{CIE}_{\mathrm{B}j}$ $(\mathrm{T}_{\mathrm{B}j}, \mathrm{CS}_{\mathrm{B}j}))$, $k \leq i \leq j \in \mathrm{N}$ }. Where N is a natural number set and k, i, and j are the number of the Common Interest Events.

If the set is empty, no secret can be shared.

D. Security Model

We think of the Blockchain as a virtual interface, HBAKE protocol can be seen as a two-party AKE in which participants are blessed by security of Blockchain. Bellare et al. introduced a formal security model for two-party password based AKE protocol, which allows a user to establish a session key with a server through a shared secret information[20]. We extended their model into a homomorphic hash and Blockchain-based one and will prove its semantic security in Section III-B.

Suppose each participant can activate multiple protocol instances and run multiple session instances in parallel. Let $\boldsymbol{P}$ denotes the protocol, $\mathrm{N}_{\mathrm{A}}^{i}$ denotes the i-th instance of node A, $\mathrm{N}_{\mathrm{B}}^{i}$ denotes the j-th instance of node B. The security is modeled by a game between a challenger $\boldsymbol{C}$ and a Probabilistic Polynomial Time (PPT) adversary $\boldsymbol{A}$. The ability of the adversary $\boldsymbol{A}$ is simulated by the following oracle queries:

Execute ($\mathrm{N}_{\mathrm{A}}^{i}$, $\mathrm{N}_{\mathrm{B}}^{i}$): This query models passive attacks of the execution of the protocol and returns a copy of the messages transmitted between $\mathrm{N}_{\mathrm{A}}^{i}$ and its partner $\mathrm{N}_{\mathrm{B}}^{i}$ during their last authentication conversation. In fact, the passive attack in the Blockchain environment is equivalent to transaction inquiry, which only threatens user's privacy.

Reveal (N^i): It models the known session key attack. $\boldsymbol{A}$ asks this query to obtain the session key held by the instance $\mathrm{N}_{\mathrm{A}}^{i}$.

Send (N^i, m): It models the active attacks against an instance. $\boldsymbol{A}$ performs this query sends a message m to instance N^i and receives a respond message in this query.

Corrupt (N^i): It models the exposure of Blockchain private key held by N. This query is used to characterize forward security.

Test (N^i): It models the semantic security of the session key. $\boldsymbol{A}$ asks this query for the fresh instance N^i only once, and challenge the tossing game $b \in \{0, 1\}$, and receives the session key if $b = 1$ or a random value with the same size if $b = 0$. The fresh instance means that an instance N^i and its partner have not been **Reveal** $(\cdot)$ query before the session key is shared.

The HBAKE security is an advantage of the PPT adversary who wins the challenge to obtain the session key after asked the above queries. Let **Succ**($\boldsymbol{A}$) denote the event that $\boldsymbol{A}$ wins, the advantage of $\mathcal{A}$ is defined as $\mathrm{Adv}_{\boldsymbol{P}}^{\mathrm{HBAKE}}(\boldsymbol{A}) = |2\Pr[\mathbf{Succ}(\boldsymbol{A})] - 1|$. The protocol $\boldsymbol{P}$ is said to be HBAKE-secure if for any PPT adversary $\boldsymbol{A}$, $\mathrm{Adv}_{\boldsymbol{P}}^{\mathrm{HBAKE}}(\boldsymbol{A})$ is negligible.

III. Homomorphic Hash and Blockchain based Authenticated Key Exchange Protocols in the Standard Model

Blockchain is a decentralized general ledger system deployed in an untrustworthy environment. Bitcoin is one of the most successful projects and our protocol works on it.

There are two types of peer nodes for this agreement: the full node that has a copy of the entire ledger and the Simplified Payment Verification (SPV) node that has only block header information. The full node verifies the validity of the transaction by retrieving the complete UTXO database between the block of the transaction and the Genesis Block while the SPV node verifies the validity of the transaction using the Merkle path of the transaction and the transaction's depth in the block[21]. Users with shared secret needs can initialize their own Common Interest Event Tag transaction (CIETX) at any time. Therefore, the full node can directly retrieve the CIETX of interest in the general ledger, and the SPV needs the Bloom filter. This paper focuses on SPV nodes.

A. Design

Suppose, user Alice intends to find someone who has a common secret with her, the **CIE** holder. She would select the well-known keywords as $\mathrm{CIET_A}$ to construct $\mathrm{CIETX_A}$ $\mathrm{Tx_{AA}}\langle \mathrm{T_{A\mathit{i}}}, \mathrm{U(U(CS_\mathit{i}))}, \mathrm{U}, \Delta\mathrm{T}\rangle$ using the OP_RETURN Script, and then publish the transaction that point to herself. Our scheme as shown in Fig. 1.

TABLE I. Summary of notations used in our scheme

Notations	**Description**
$\mathrm{Tx_{AB}}$, $\mathrm{Tx_{BA}}$	Transactions between user Alice and Bob
T_k	Tag of k-th User's Common Interest Event
CS_k	Common string of k-th User's Common Interest Event
$\mathrm{U}(\cdot)$	$\mathrm{U}(\cdot) \in \mathcal{U}\{\mathrm{U}(\cdot) \| (0,1)^* \to (0,1)^l\}$
$\mathrm{H}(\cdot)$	The homomorphism hash function defined in section II-A
$\Delta\mathrm{T}$	Maximum interaction delay
δ	Blockchain time less than $\Delta\mathrm{T}$
m	Any plaintext
key	Session key between Alice and Bob
→, ←	Data flow direction
$\|\|$	The concatenation operation

After that, the details of the scheme are presented. Some notations are used to describe the scheme clearly, as shown in Table I.

The first stage: Looking for volunteers of the Common Interest Group.

Step1: Bob selects some well-known keywords as $\mathrm{CIET_B}$ to initialize Bloom filter and sends it to other peers, if he receives some results of the Bloom filter match from another peer, it means that someone on the network probably hold the same secrets as him. He would abort the protocol if validation fails. Otherwise, key exchange authentication is performed.

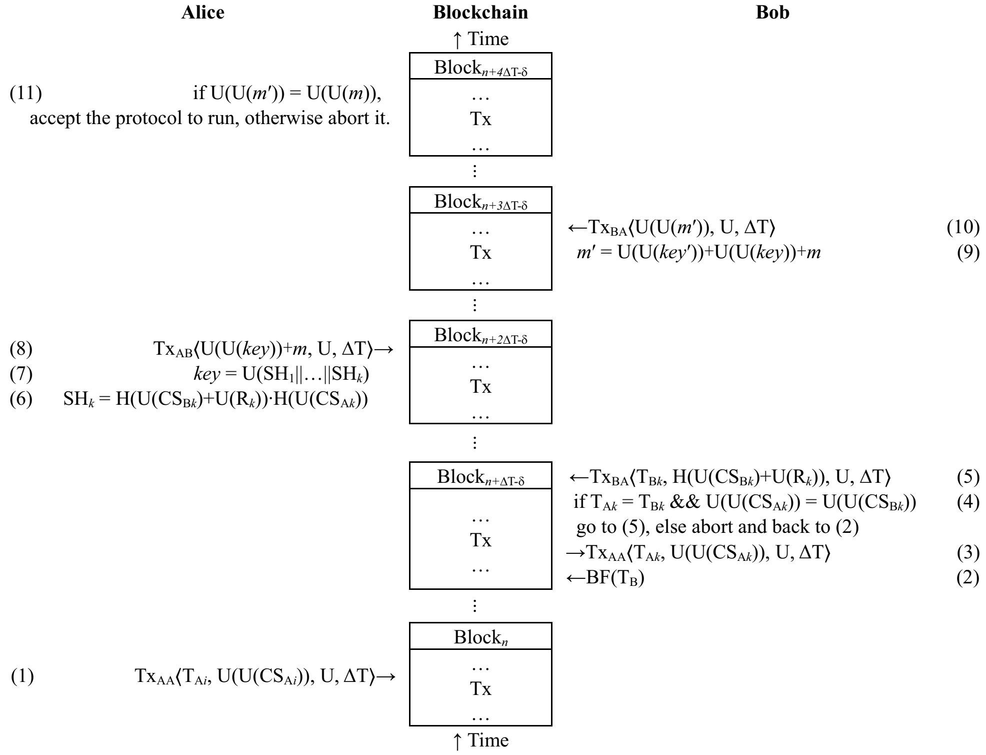
Alice
Blockchain
Bob
↑ Time
$\mathrm{Block}_{n+4\Delta\mathrm{T}-\delta}$
(11) if $\mathrm{U(U(\mathit{m}'))} = \mathrm{U(U(\mathit{m}))}$,
accept the protocol to run, otherwise abort it.
… Tx …
$\mathrm{Block}_{n+3\Delta\mathrm{T}-\delta}$
$\leftarrow \mathrm{Tx_{BA}}\langle \mathrm{U(U(\mathit{m}'))}, \mathrm{U}, \Delta\mathrm{T}\rangle$ (10)
$m' = \mathrm{U(U(\mathit{key}'))} + \mathrm{U(U(\mathit{key}))} + m$ (9)
$\mathrm{Block}_{n+2\Delta\mathrm{T}-\delta}$
(8) $\mathrm{Tx_{AB}}\langle \mathrm{U(U(\mathit{key}))} + m, \mathrm{U}, \Delta\mathrm{T}\rangle \to$
(7) $key = \mathrm{U}(\mathrm{SH_1} \| \ldots \| \mathrm{SH}_k)$
(6) $\mathrm{SH}_k = \mathrm{H}(\mathrm{U}(\mathrm{CS}_{\mathrm{B}k}) + \mathrm{U}(\mathrm{R}_k)) \cdot \mathrm{H}(\mathrm{U}(\mathrm{CS}_{\mathrm{A}k}))$
$\mathrm{Block}_{n+\Delta\mathrm{T}-\delta}$
$\leftarrow \mathrm{Tx_{BA}}\langle \mathrm{T}_{\mathrm{B}k}, \mathrm{H}(\mathrm{U}(\mathrm{CS}_{\mathrm{B}k}) + \mathrm{U}(\mathrm{R}_k)), \mathrm{U}, \Delta\mathrm{T}\rangle$ (5)
if $\mathrm{T}_{\mathrm{A}k} = \mathrm{T}_{\mathrm{B}k}$ && $\mathrm{U}(\mathrm{U}(\mathrm{CS}_{\mathrm{A}k})) = \mathrm{U}(\mathrm{U}(\mathrm{CS}_{\mathrm{B}k}))$ (4)
go to (5), else abort and back to (2)
$\to \mathrm{Tx_{AA}}\langle \mathrm{T}_{\mathrm{A}k}, \mathrm{U}(\mathrm{U}(\mathrm{CS}_{\mathrm{A}k})), \mathrm{U}, \Delta\mathrm{T}\rangle$ (3)
$\leftarrow \mathrm{BF}(\mathrm{T_B})$ (2)
Block_n
(1) $\mathrm{Tx_{AA}}\langle \mathrm{T}_{\mathrm{A}i}, \mathrm{U}(\mathrm{U}(\mathrm{CS}_{\mathrm{A}i})), \mathrm{U}, \Delta\mathrm{T}\rangle \to$
↑ Time

Fig. 1. An illustration of the details of our scheme

The second stage: Key exchange.

Step2: Bob find a volunteer of the Common Interest Group, he would hash the each $\mathrm{CS_B}$ which corresponds to the tag he received with the hash function U, and determine whether $\mathrm{T}_{\mathrm{A}k} = \mathrm{T}_{\mathrm{B}k}$ && $\mathrm{U(U(CS}_{\mathrm{A}k})) = \mathrm{U(U(CS}_{\mathrm{B}k}))$ holds, if yes, select k random string R_k of length equal to $\mathrm{CS}_{\mathrm{B}k}$, and use the tuple $\langle \mathrm{T}_{\mathrm{B}k}, \mathrm{H(U(CS}_{\mathrm{B}k})+\mathrm{U(R}_k)), \mathrm{U}, \Delta\mathrm{T}\rangle$ filling OP_RETURN to construct a transaction that point to Alice and publish it.

Step3: Alice will detect the transaction, and abort the protocol if validation fails. Otherwise, she would calculate each secret hash value $\mathrm{SH}_k = \mathrm{H(U(CS}_{\mathrm{B}k})+\mathrm{U(R}_k))\cdot\mathrm{H(U(CS}_{\mathrm{A}k}))$ which corresponds to each common Tag. If Bob is the holder of common secret, there must be $\mathrm{SH}_k = \mathrm{H(U(R}_k))$.

At this point, all participants believe with confidence the CS comes from someone who knows his secret or have a common secret with him. The common secret is $key = \mathrm{U(SH_1}||\ldots||\mathrm{SH}_k)$ which is used for subsequent secure communications.

The third stage: Non-essential authentication.

Step4: For fairness and security, Alice initiates the authentication process. She will publish the transaction $\mathrm{Tx_{AB}}\langle \mathrm{U(U(}key))+m, \mathrm{U}, \Delta\mathrm{T}\rangle$ that point to Bob, m is a random string of length equal to $\mathrm{U(U(}key))$.

Step5: Bob will detect the transaction, and abort the protocol if validation fails. Otherwise, he would calculate the $m' = \mathrm{U(U(}key'))+\mathrm{U(U(}key))+m$ and publish the transaction $\mathrm{Tx_{BA}}\langle \mathrm{U(U(}m')), \mathrm{U}, \Delta\mathrm{T}\rangle$ that point to Alice.

Step6: Alice will detect the transaction, and abort the protocol if validation fails. Otherwise, she would determine whether $\mathrm{U(U(}m'))$ and $\mathrm{U(U(}m))$ are equal, and if so accept the protocol to run, otherwise abort it.

Non-essential authentication avoids protocol failure due to impersonation attacks. However, since each valid transaction on the Blockchain is certified for payment and the transactions associated with the protocol instance are consistent and traceable, impersonation attacks are less of a threat to this protocol.

B. Syntax

This section proves that the proposed scheme in Section III-A. is HBAKE-secure under the security model defined in Section II-D. We only discuss the semantic security of the scheme. Another element of the adversary transaction fees cost as a security guarantee is provided by the Blockchain.

Theorem 1 Let $\boldsymbol{P}$ be the proposed protocol, $\boldsymbol{A}$ is a PPT adversary who makes at most q_s active attacks. Under the assumptions of one-way hash function and Blockchain transaction-level security, the semantic security advantage of $\boldsymbol{A}$ breaking $\boldsymbol{P}$ is at most:

$$\mathrm{Adv}_{\boldsymbol{P}}^{\mathrm{HBAKE}}(\boldsymbol{A}) \le {q_s^2}/{2^{l-1}} ,$$

where l is security parameters.

Proof: Let S_i be the event that the adversary $\boldsymbol{A}$ wins game G_i, namely, $\boldsymbol{A}$ guessed the correct b. The game sequence is described as follows.

Game G_0: It is the original attack game defined in Section II-D. According to definitions, we have

$$\mathrm{Adv}_{\boldsymbol{P}}^{\mathrm{HBAKE}}(\boldsymbol{A}) = |2\mathbf{Pr}[\mathbf{S_0}] - 1|.$$

Game G_1: This game models a passive attack by querying the **Execute** $(\cdot)$ oracle. But the adversary $\boldsymbol{A}$ can hardly increase the advantage of winning the game. Since the secret key is directly related to random numbers R_k and indirectly to common strings $\mathrm{CS_k}$, the $\boldsymbol{A}$ can only get $\mathrm{H(U(CS}_k) + \mathrm{U(R}_k))$ and $\mathrm{U(U(CS}_k))$. Under the assumption of one-way hash function, the adversary $\boldsymbol{A}$ cannot get more advantages than the tossing game. Thus we have $\mathbf{Pr}[\mathrm{S_1}] = \mathbf{Pr}[\mathrm{S_0}]$.

Game G_2: We transfer game G_1 to this game by adding the **Send**$(\cdot)$ query to model an active attack. Because fake **step** 3 and subsequent transactions fail to pass the Blockchain consensus authentication, the adversary $\boldsymbol{A}$ has to launches **Send**$(\cdot)$ query from **step** 2. $\boldsymbol{A}$ sends faked transactions $\mathrm{Tx_F}\langle \mathrm{T}_k, \mathrm{H(U(CS}_k')+\mathrm{U(R}_k)), \mathrm{U}, \Delta\mathrm{T}\rangle$to the honest user simulated by the challenger $\boldsymbol{C}$. There must be $\mathrm{SH}_k' = \mathrm{H(U(R}_k'))$ and $key' = \mathrm{U(SH_1}'||\ldots||\mathrm{SH}_k')$. If and only if $key' = key$, the protocol can continue to run. According to the collision free of hash function and the birthday paradox, we have

$$|\mathbf{Pr}[\mathbf{S_2}]\text{-}\mathbf{Pr}[\mathbf{S_1}]| \le {q_s^2}/{2^{l-1}} .$$

Game G_3: We transfer game G_2 to this game by adding the **Corrupt**$(\cdot)$ query to enhance attack ability of the adversary $\boldsymbol{A}$. That is, $\boldsymbol{A}$ obtains the honest user's Blockchain private key, which has the ability to forge a valid transaction. Although the adversary $\boldsymbol{A}$ can forge all valid transactions at step 2 and beyond, due to the collision free of hash function, $\boldsymbol{A}$ cannot gain more advantages than game G_2. So we have $\mathbf{Pr}[\mathrm{S_3}] = \mathbf{Pr}[\mathrm{S_2}]$.

From all the above equations, we have:

$$\mathrm{Adv}_{\boldsymbol{P}}^{\mathrm{HBAKE}}(\boldsymbol{A}) \le {q_s^2}/{2^{l-1}} .$$

Theorem 1 proved.

C. Discussion on Possible Attacks

We discussed the security of the HBAKE scheme by analyzing passive attacks and some active attacks in Section III-B. Next we discuss other possible attacks that this scheme can resist.

Man-in-the-middle Attack (MitmA). In our scheme, the condition that the participants successfully negotiate the session key is that the CIE verifies each other and the transaction containing the CIE passes the Blockchain global consensus authentication. We aware that it is infeasible for the adversary $\boldsymbol{A}$ to produce a valid transaction which can pass the CIE verification of the participants and global consensus authentication, even if $\boldsymbol{A}$ can get the private key and create a valid signatures to pass the consensus authentication, but can not pass the CIE verification. Moreover, the MitmA on forged transactions disrupt the consistency of the transactions between the parties are easily monitored. The above analysis shows that our scheme can resist the MitmA.

Replay Attack (RA). It is hard for the adversary $\boldsymbol{A}$ to replay the transaction in the P2P network since current timestamps are used in each transaction and block. And each CIETX contains the maximum delay ΔT allowed by the initiator. So our protocol is immune to RA.

Denial-of-service Attack (DosA). There are two possible DosA in this scenario, information black holes and massive malicious intrusions. It is unlikely that an adversary $\boldsymbol{A}$ will play a black hole of information in a decentralized network to implement a DosA. Blockchain consensus mechanisms and transaction fees mechanisms can effectively prevent adversaries from implementing such high-cost attack. In addition, the use of high-entropy CIES and multi-factor CIES is also an effective way to prevent denial of service attacks.

D. Performance Analysis

1) Security Features

In our scheme, Blockchain provides transaction-level security, and a one-way hash function ensures that shared secrets do not leak.

Mutual Authentication. The protocol participants use the public-private key pairs of Blockchain users and their derived addresses and signatures to authenticate CIE holders, and the Blockchain transaction consensus authentication mechanism ensures the integrity of CIETX.

Privacy Protection. A non-real-life identity used in the network can effectively protect user privacy. Bitcoin uses the public key derived address to identify users at the transaction level. Such pseudonym technique does not prevent transaction analysis and user privacy links. Our scheme can be anonymized using Tor[22] technology.

2) Computational Complexity

We do not consider Bitcoin's computational overhead and transaction fees, and only briefly analyze the computational complexity of this protocol.

Assume that there are k CIEs between protocol instances, the cost of the U(·) is T_u, the cost of the H(·) is T_h, the cost of the string matching is T_c, and the cost of the l-bit XOR operation is T_{xor}. As shown in Fig. 1, except for the advertising step (1), the bloom filter query step (2) and the step (8)-(11), the calculation cost for successfully negotiating the session key is $(2T_c+5T_u+2T_h+2T_{xor})k + T_u$.

3) Performance comparison

According to our study, HBAKE's research is rare. We select two typical HBAKE protocols and compare the performance of the protocol proposed in this paper from four aspects: security model, security assumption, authentication strategy, and extensibility.

In comparison, we use RO to represent a random oracle model, use ST to represent a standard model, use BTLS to indicate Blockchain transaction-level security, and use OW to represent hash function one-way security. As can be seen from Table II, only the protocol mentioned in this paper is certifiable security under the standard model. Compared with McCorry's [16] protocol, our protocol not only guarantees security under the standard model, but also supports offline operation of the protocol, which makes the protocol more flexible and scalable. Compared with Bui's [17], our protocol is more secure.

TABLE II. COMPARISONS OF HBAKE PROTOCOLS

Protocols	Security Model	Security Assumption	Authentication Strategy	Support Offline
McCorry1[16]	RO	CDH	ECDSA	N
McCorry2[16]	RO	CDH	ECDSA	N
Bui[17]	—	BTLS	ECDSA	Y
Our	ST	OW/DL/BTLS	ECDSA	Y

IV. TRUST BOOTSTRAPPING BETWEEN STRANGERS IN SOCIAL NETWORK

Our scheme is mainly applied to strangers bootstrapping trust scenario, such as the use of DNA to securely find their biological relatives.

Assuming that Alice and Bob have parental relations, they have been strangers for years separated by war. Right now, they want to find each other on the promise of privacy. Then, Alice selects some tag strings, e.g. well known DNA polymorphic markers (D21S11, D7S820, D8S1179, CSF1PO, D3S1358, TH01, D13S317, D16S539, D2S1338 and D19S433). And use these strings to construct the following 10 transactions that point to herself and publish it.

$$Tx_{AA01}\langle D21S11, U(U(D21S11_{A1})), U(U(D21S11_{A2})), U, \Delta T\rangle,$$
$$\dots$$
$$Tx_{AA10}\langle D7S820, U(U(D7S820_{A1})), U(U(D7S820_{A2})), U, \Delta T\rangle.$$

The first stage: Looking for volunteers of the Common Interest Group.

Step1: Bob selects D21S11, D7S820, TH01, D13S317, and D19S433 as CIETB to initialize Bloom filter and sends it to other peers, if he receives some results of the Bloom filter match from another peer, it means that someone on the network probably hold the same secrets as he. He will abort the protocol if validation fails. Otherwise, key exchange authentication is performed.

The second stage: Key exchange.

Step2: Bob find a volunteer of the Common Interest Group, he will hash the each CS_B which corresponds to the tag he received with the hash function U, and determine whether the hash U(U(·)) corresponding to the short tandem repeats matches at least one. Let us suppose:

- D21S11: $U(U(D21S11_{B2})) = U(U(D21S11_{A2}))$,
- D7S820: $U(U(D7S820_{B2})) = U(U(D7S820_{A1}))$,
- TH01: $U(U(TH01_{B2})) = U(U(TH01_{A2}))$,
- D13S317: $U(U(D13S317_{B2})) = U(U(D13S317_{A2}))$,
- D19S433: $U(U(D19S433_{B2})) = U(U(D19S433_{A1}))$.

Bob will select 5 random strings $R_1 \dots R_5$ of length equal to CS_{Bk}, and construct the following 5 transactions that point to Alice and publish it.

$$Tx_{BA01}\langle D21S11, H(U(D21S11_{B2})+U(R_1)), 22, U, \Delta T\rangle,$$
$$\dots$$
$$Tx_{BA05}\langle D19S433, H(U(D19S433_{B2})+U(R_5)), 21, U, \Delta T\rangle.$$

Step3: Alice will detect the transaction, and abort the protocol if validation fails. Otherwise, she will calculate each secret hash value which corresponds to each common Tag.

$SH_1 = H(U(D21S11_{B2})+U(R_1))\cdot H(U(D21S11_{A2}))$,

$$\ldots$$

$SH_5 = H(U(D19S433_{B2})+U(R_1))\cdot H(U(D19S433_{A1}))$.

According to the assumptions in step2, there must be

$$SH_1 = H(U(R_1)),$$

$$\ldots$$

$$SH_5 = H(U(R_5)).$$

At this point, all participants believe with confidence the CIES comes from someone who knows his secret or have a common secret with him. The common secret is key = $U(SH_1\|\ldots\|SH_5)$ which is used for subsequent secure communications.

V. Conclusion

In this paper, we have designed a homomorphic hash and Blockchain based authenticated key exchange protocol with privacy protection, and prove its security under the standard model based on hash one-way, discrete logarithm and Blockchain transaction-level security assumption, and discussed the attack that the proposed scheme can resist. Compared with the existing HBAKE protocol, our protocol is more secure and more flexible.

Acknowledgment

This work is supported by National Natural Science Foundation of China(No. 61562077 and No. 61772022), the Lanzhou City University Young Teachers Research Project No. LZCU-QN2016-11. Many thanks go to the anonymous reviewers for their detailed comments and suggestions.

References

[1] W. Sherchan, S. Nepal, and C. Paris, “A Survey of trust in social networks,” ACM Comput. Surv. 45, 4, Article 47, 2013, 33 pages.

[2] S. M. Bellovin, M. Merritt, “Encrypted key exchange: Password-based protocols secure against dictionary attacks,” IEEE Proceedings of the Computer Society Symposium on Research in Security and Privacy, 1992, pp. 72-84.

[3] V. Boyko, P. MacKenzie, S. Patel, “Provably secure password-authenticated key exchange using Di-Hellman,” International Conference on the Theory and Applications of Cryptographic Techniques, 2000, pp. 156-171.

[4] W. Diffie, P. C. Van Oorschot, M. J. Wiener, “Authentication and authenticated key exchanges,” Designs, Codes and cryptography 2(2), 1992, pp. 107-125.

[5] F. Hao, P. Ryan, “J-PAKE: Authenticated key exchange without PKI,” Trans-actions on computational science XI, 2010, pp. 192-206.

[6] D. P. Jablon, “Strong password-only authenticated key exchange,” ACM SIGCOMM Computer Communication Review 26(5), 1996, pp. 5-26.

[7] T. Wu, “The secure remote password protocol,” Proceedings of the Internet Society Symposium on Network and Distributed System Security. vol. 98, 1998, pp. 97-111.

[8] T. Dierks, “The transport layer security (TLS) protocol version 1.2,” RFC 5246, 2008.

[9] A. Shamir, “Identity-based cryptosystems and signature schemes,” Workshop on the Theory and Application of Cryptographic Techniques, 1984, pp. 47-53.

[10] H. Kalodner, M. Carlsten, P. Ellenbogen, J. Bonneau, A. Narayanan, “An empirical study of Namecoin and lessons for decentralized namespace design,” Proceedings of the Workshop on the Economics of Information Security (WEIS), 2015.

[11] M. S. Melara, A. Blankstein, J.Bonneau, E. W. Felten, M. J. Freedman, “CONIKS: Bringing key transparency to end users,” Proceedings of the USENIX Security Symposium, 2015, pp. 383-398.

[12] J. Yu, M. Ryan, C. Cremers, “DECIM: Detecting endpoint compromise in messaging,” Tech. rep, 2015.

[13] W. Dong, V. Dave, L. Qiu, “Secure friend discovery in mobile social networks,” INFOCOM’11, Shanghai, China, Apr. 2011.

[14] V. Miller, “Use of Elliptic Curves in Cryptography,” Advances in Cryptology-CRYPTO85 Proceedings, 1986, pp. 417-426.

[15] F. Hao, “On robust key agreement based on public key authentication,” Financial Cryptography and Data Security, 2010, pp. 383-390.

[16] P. McCorry, S. F. Shahandashti, D. Clarke, F. Hao, “Authenticated Key Exchange over Bitcoin,” L. Chen, S. Matsuo (eds) Security Standardisation Research. Lecture Notes in Computer Science, vol 9497, 2015.

[17] T. Bui, T. Aura, “Key Exchange with the Help of a Public Ledger,” F. Stajano, J. Anderson, B. Christianson, V. Matyáš (eds) Security Protocols XXV. Security Protocols 2017. Lecture Notes in Computer Science, vol 10476, 2017.

[18] M. N. Krohn, M. J. Freedman, and D. Mazieres,"On-the-fly verification of rateless erasure codes for efficient content distribution," in Proc. IEEE Symp. Secur. Privacy (S&P), May 2004, pp. 226–240.

[19] S. Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System,” 2008.

[20] M. Bellare, D. Pointcheval, and P. Rogaway, "Authenticated key exchange secure against dictionary attacks," In: Advances in Cryptology - Eurocrypt 2000. vol. 1807, B. Preneel, Ed., ed Berlin: Springer-Verlag Berlin, 2000, pp. 139-155.

[21] G.O. Karame, E. Androulaki, and S. Capkun, “Double-spending fast payments in bitcoin”. In Proceedings of the 2012 ACM conference on Computer and communications security, 2012 pp. 906-917.

[22] Tor Projects. https://www.torproject.org/about/overview.html.en. Accessed on 2018-02-06.