# A Model of Decentralized Social Internet of Things using Blockchain Offline Channels

Subhasis Thakur
*National University of Ireland, Galway*
Galway, Ireland
subhasis.thakur@nuigalway.ie

John G. Breslin
*National University of Ireland, Galway*
Galway, Ireland
john.breslin@nuigalway.ie

***Abstract*—Social Internet of Things (SIoT) enables IoT devices to discover the services offered by each other using a social network. The social network connects two devices if they have a common owner or both manufactured by the same company or they are situated at the same location etc. Centralized SIoT platforms have security and privacy problems. We advance the state of art in SIoT as we have developed a decentralized SIoT platform. In a decentralized SIoT platform, the services provided by a device is only known to its neighbours. Due to such characteristics of decentralized SIoT platform, there are several challenges in designing it, such as verification of social neighbourhood, privacy-preserving search in SIoTs, and scalability of SIoT platform. In this paper, we propose a blockchain-based SIoT platform to mitigate these problems. Our main contributions are: We developed a decentralized SIoT platform that allows secure privacy-preserving method to verify social neighbourhood a device, we developed a protocol to find devices in the decentralized SIoT and we developed a high scale decentralized SIoT platform using blockchain offline channels. We prove the proposed decentralized SIoT platform is privacy-preserving, secure, and scalable.**

***Index Terms*—Social Internet of Things, Blockchains, Offline channels**

## I. Introduction

Social Internet of Things (SIoT) [1], [2] exploits a social structure among the IoT devices using co-ownership information of the devices, social network among the device owners, common manufacturers of the devices, and common location of the devices. In the state of the art, the SIoT platform is centralized as a new device can register itself to the platform, form a social neighbourhood and search the network created by the social neighbourhood to find another device that is providing a particular service. The state of the art research in IoT have investigated protocols to register an IoT device to the SIoT platform, algorithms to form efficient neighborhood and algorithm to search.

However, a lack of trust in the centralized SIoT platform and privacy issues may prevent participation in the SIoT platform. A decentralized SIoT platform is suitable to mitigate these trust issues. In such a decentralized SIoT platform, only a device knows its social neighbours. In this paper, we propose to use blockchains to build a decentralized SIoT platform. However, building a blockchain-based SIoT platform has the following challenges:

- **Verification of the social neighbourhood:** In a decentralized SIoT platform, a device's neighbourhood is only known to the device. The lack of knowledge of SIoT network topology creates the problem of verifying the existence of paths between two devices. A malicious device may use false social neighbourhood information.
- **Privacy preserving search:** A primary objective of SIoT is to find a device providing a particular service [3]. In a centralized SIoT platform, devices register the service they provide to a centralized entity. However, in a decentralized SIoT platform, the services provided by a device is only known to its neighbours. The challenge in a decentralized SIoT platform is to develop a distributed search algorithm which do not reveal a social neighbourhood and which hides the identity information of the querying device and the result of the search.
- **Scalability:** Scalability of decentralized SIoT platform is an issue. A blockchain-based SIoT platform may inherit the blockchain's scalability problems.

Our main contributions are as follows:

1) We developed a blockchain offline channel-based SIoT platform which is highly scalable as the offline channels significantly reduce the number of transactions needed to be recorded in the blockchain. We improve the state of art [1], [4] in SIoT by providing a scalable blockchain-based platform.
2) We developed a secure and privacy-preserving social neighbourhood formation protocols using blockchain offline channels. This social neighbourhood formation protocol is secure as a device may not wrongfully claim to be owned by a device owner. This protocol is privacy-preserving as only a device knows its neighbours. Two devices may be neighbours of each other but our protocol ensures that they will not know remaining members of each other's neighbours. We improve the state of art in SIoT navigability [3], [7] problem by developing a secure and privacy-preserving social neighbourhood

This publication has emanated from research supported in part by a research grant from Science Foundation Ireland (SFI) and the Department of Agriculture, Food and the Marine on behalf of the Government of Ireland under Grant Number SFI 16/RC/3835 (VistaMilk), and also by a research grant from Science Foundation Ireland (SFI) under Grant Number SFI/12/RC/2289_P2 (Insight), with both grants co-funded by the European Regional Development Fund.

formation protocol.

3) We developed a secure and privacy-preserving data exchange protocol between two devices who are not in each other social neighbourhood. This protocol is secure as it can authenticate ownership of a device. The protocol ensures that two devices may only engage in using services provided by each other if there exists a proper path in the social network between them. The protocol ensures that the social neighbourhood of these two devices is not revealed during such a data exchange between them. We improve upon existing models of trust computation [11]–[13] among two devices unknown to each other by allowing two devices to evaluate and establish trust in each other in a secure and privacy-preserving manner.

The paper is organized as follows: in section 2 we discuss related literature, in section 3 we describe the decentralized SIoT platform built with blockchain offline channels, in section 4 we present security and privacy analysis fo the SIoT platform, and we conclude the paper in section 5.

## II. Related Literature

[4] proposed a hierarchical structure of blockchains to be used in resource-constrained IoT devices. [7] discusses security and privacy challenges for IoTs. [3] analyses the navigation problem in social IoTs which aims to find a path in the social network among the IoT devices to find an IoT with a specific capability. [8] also investigates navigation problems in social IoT networks. [9] provides a location-aware service discovery algorithm for social IoTs. [10] provides security analysis IoT protocols. [11] proposed a trust computation algorithm for Social IoTs. [12] provides a decentralised service discovery algorithm for IoTs. [13] provides a decentralised algorithm for trust evaluation in Vehicular IoTs. [14] provided a survey on trust management in IoTs. Bitcoin lightning network was proposed in [15] which allows peers to create and transfer funds among them without frequently updating the blockchain. Similar networks are proposed for Ethereum (Raiden network). [6] proposed a landmark-based routing protocol for fund transfer in a credit network. [5] enhanced the landmark-based routing algorithm developed in [6] by reducing the path length for PBT execution. Our contributions advance the state of the art in SIoT with a novel method to create a secure, scalable, and privacy-preserving SIoT platform.

## III. SIoT network using Blockchain Offline Channel

In this section, we will explain how to use blockchain offline channels to create SIoT platform.

### A. Blockchain Offline Channels

Blockchain offline channels [15] uses multi-signature addresses to open an offline channel among peers of the blockchain. As shown in figure 1 the procedure to open a bi-directional offline channel between two peers $A$ and $B$ is as follows:

- $A$ and $B$ need two multi-signature addresses $M_{A,B}, M'_{A,B}$ between them to establish an offline channel. These addresses need a signature from both to transfer funds from it.
- First $A$ will create a random string $Key_A$ and produce its Hash $Lock_B$ (SHA246). $A$ will send $B$ $Lock_A$. $B$ will do the same.
- Next, $A$ will create a Hashed Time Locked Contracts[1] (HTLC) (a new transaction) as follows:
  - Inputs to the new HTLC ($HTLC_A$) created by $A$ are unspent transactions funding $M_{A,B}$.
  - Say both peer funds 1 token to $M_{A,B}$. The HTLCs are designed to ensure that each party gets 1 token if they do not change their agreement on sharing the total fund.
  - $HTLC_A$ states that, from $M_{A,B}$ 1 token will be sent to $A$ and 1 token to another multi-signature address of $M'_{A,B}$. From $M'_{A,B}$ all tokens will be sent to $B$ after 10 days (maybe calculated as the number of new blocks to be added to the blockchain) if $A$ does not claim these tokens before 10 days by producing $Key_B$.
  - Similarly, the HTLC created by $B$, $HTLC_B$ states that, from $M_{A,B}$ 1 token will be sent to $B$ and 1 token to another multi-signature address of $M'_{A,B}$. From $M'_{A,B}$ all tokens will be sent to $A$ after 10 days (maybe calculated as the number of new blocks to be added to the blockchain) if $B$ does not claim these tokens before 10 days by producing $Key_A$.
  - After receiving these HTLCs both parties send 1 token to $M_{A,B}$.
- If $A$ and $B$ want to change the share of $M_{A,B}$ then, new HTLCs will be exchanged. In order to do so, both peers first reveal their old keys ($Key_A, Key_B$). The offline channel is opened once funds are sent to $M_{A,B}$ and it is closed once an HTLC is published to the blockchain network.

The security problem of the above offline channels are as follows:

- It is possible that a peer may publish an old HTLC to the blockchain network. For example, it is possible that the current agreed sharing of $M_{A,B}$ is .9 tokens should go $A$ and 1.1 should go to $B$. But there exists an old $HTLC_A$ which states that both should get 1 token each. Consider $HTLC_A$ is the same as defined as before.
- If $B$ publishes $HTLC_A$ then $A$ immediately gets 1 token, remaining tokens go to $M'_{A,B}$ where $B$ waits for 10 days to get 1 token. However, $A$ can observe this HTLC in the blockchain and can claim all tokens from $M'_{A,B}$ by providing $Key_B$. Note that as $HTLC_A$ is an old HTLC, $A$ must have $Key_B$ to create the new HTLCs.

The privacy problems for the above offline channel are as follows: It is possible to identify the existence of offline

[1] https://en.bitcoin.it/wiki/Hash_Time_Locked_Contracts

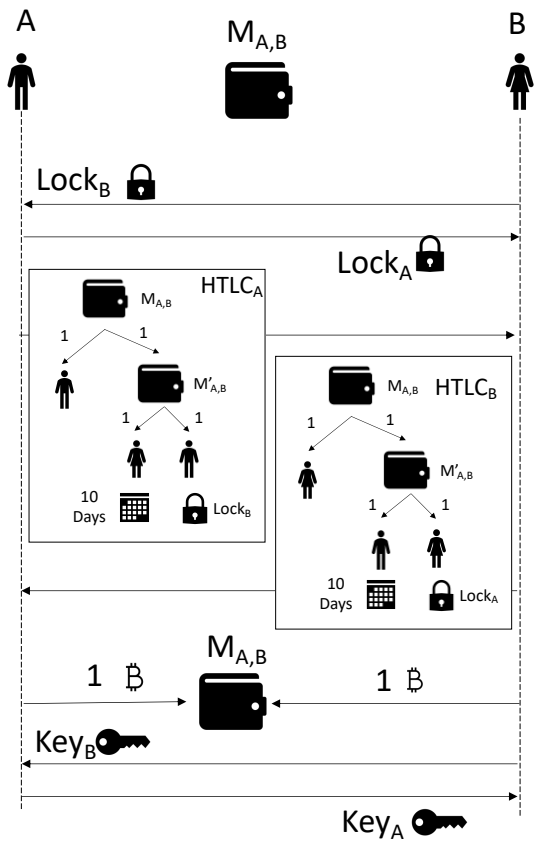


Fig. 1. Procedure of creating offline channels.

channels between any pair of peers as the multi-signature address may be visible by all peers and transactions to and from such multi-signature addresses are also visible for all peers.

### B. Blockchains and IoT network

It will be assumed that the IoT devices are too computationally challenged to operate the blockchain node. A group of IoT devices will be part of an IoT network that will contain one 'gateway' node, referred to as 'agent' which will have sufficient computation capability to operate a blockchain wallet. A blockchain wallet will enable a peer to create and receive transactions. In this architecture (shown in figure 2) of connecting IoT networks we assume the following:

1) Peers in the blockchain network can not directly communicate via the blockchain (by sending transactions) with the sensors on any IoT network.
2) An agent of a blockchain network can communicate with each sensor of its IoT network. It is assumed that an adversary can not take control of the agent and can not disrupt the communications in each IoT network.
3) An IoT device may be controlled by an adversary.
4) The blockchain network is secure, i.e., it is not possible to overwrite transactions of the blockchain network.

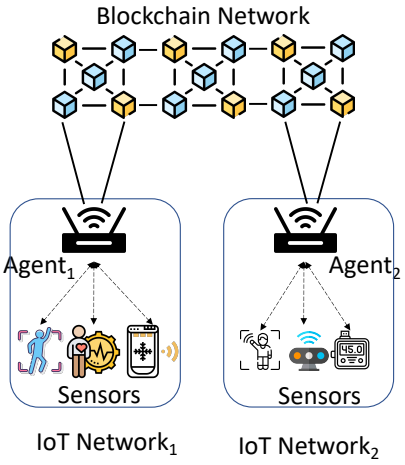


Fig. 2. Blockchain network and IoT networks.

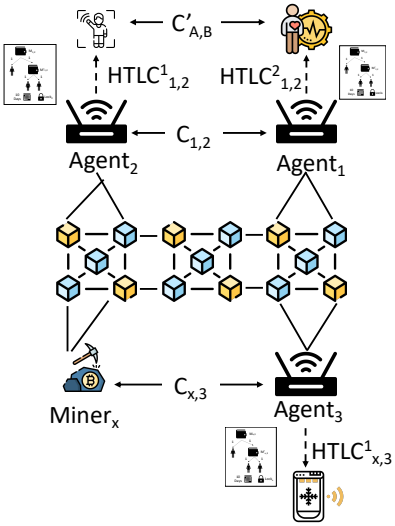


Fig. 3. Channel constructions among devices. A device will use the gateway node (who can access the blockchain) to verify the information provided by another device.

### C. Social Neighbourhood: Offline channel among Device Owners

We construct offline channels among sensors (shown in figure 3) belonging to different IoT networks to facilitate decentralised social neighbourhood formation. Two devices can form a decentralised social neighbourhood if the following holds:

- Two devices can verify the authenticity of the devices by verifying the identity of their respective owners and by verifying such ownership.
- Two devices can secure their social neighbourhood by encrypting their communications with encryption keys only known by them.

- The social association between two devices, i.e., they communicate with encryption keys only known by themselves should not be identified by any adversary.

We will first construct a new type of offline channels which will register the encryption keys to be used by the devices. Owner of IoT networks represented by 'agents' of two IoT networks may open such a channel (shown in figure 4) as follows:

1) Two peers $A$ and $B$ exchanges a set of locks $(Lock_A^1, \ldots, Lock_A^n)$ and $(Lock_B^1, \ldots, Lock_B^n)$. They also exchange tree structures for both sets of locks. The tree is such that each node except the leaf nodes have exactly $k_1$ children and there are $k_2$ levels. Hence $n = k_1 + (k_1)^2 + (k_1)^3 + \cdots + (k_1)^{k_2} = k_1(1-k_1^{k_2})/(1-k_1)$.
2) The $HTLC_A$ from $A$ to $B$ states the following:
   a) $A$ will get 1 token immediately from $M_{A,B}$. Other token will be given to $M'_{A,B}$.
   b) $B$ will get 1 token from $M'_{A,B}$ after 10 days if $A$ does not claim these tokens before 10 days by producing a set of keys along one shortest path of the above-mentioned tree over the locks from any leaf to the root. Such a sequence of keys should also accompany a set of time-stamped receipts of keys from $A$ such that each time-stamped lock will prove that the key was received before the time deadline for the lock in $HTLC_A$.
   c) The protocol shown in the next section, explains the time-stamped receipts of the keys.
3) Similarly, $B$ constructs and sends $HTLC_B$ to $A$.
4) To update the channel balance, both parties should exchange a set of keys along with one shortest path of the above-mentioned tree over the locks from any leaf to the root.

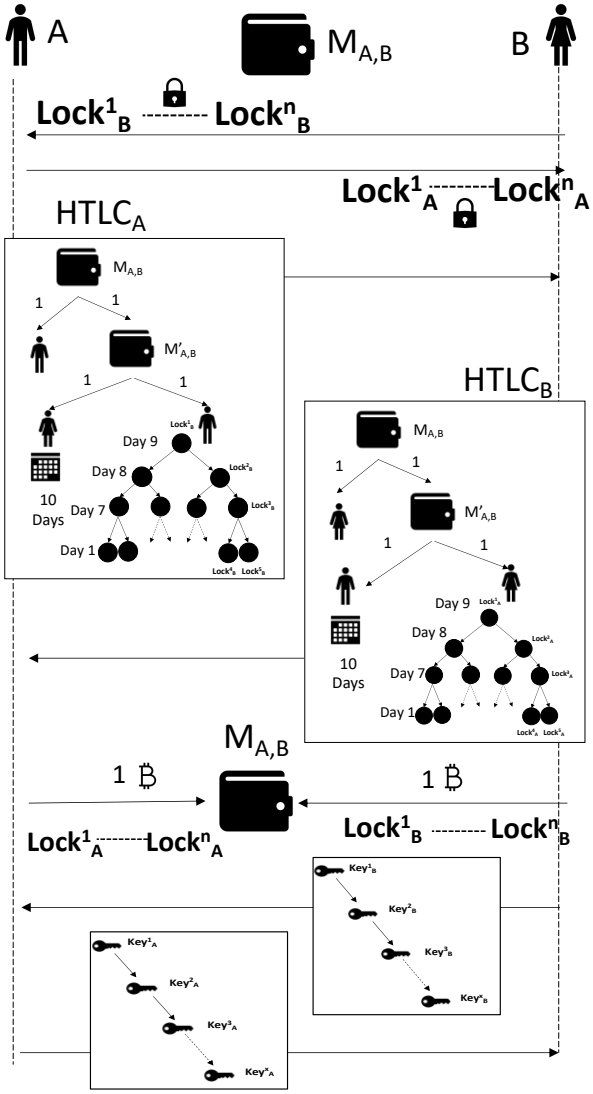


Fig. 4. Channel constructions among agents or between one agent and a miner of the blockchain network. It allows an agent (gateway node of an IoT network) to register encryption keys to be used by devices in the IoT network controlled by the agent in the blockchain. Other devices can verify the authenticity of another device by checking the existence of the Hash of the encryption key in the blockchain via its agent.

The time-stamped tree over the locks in each HTLC will be used by a device owner to distribute encryption keys among each IoT device it owns which will be part of the SIoT platform. As shown in figure 4 and 5, there are $k1$ sub-trees from each root of the tree over the locks in an HTLC. Keys corresponding to one sub-tree may be allocated to one IoT device. Such a device will sequentially use these keys starting from the keys corresponding locks in the leaf nodes of one such sub-tree. Note that leaf nodes have the lowest expiry time as mentioned in the HTLCs, hence these must be used first. The expiry time of keys is gradually increased with the level of the tree. In the next section, we will explain how two IoT devices can use such encryption keys as they form a social neighbourhood.

### *D. Social Neighbourhood: Offline Channel among Devices*

Two IoT devices $D_A$ and $D_B$ who belong to IoT networks with agents $A$ and $B$ respectively can create a social neighbourhood between themselves by following this procedure (shown in figure 6):

1) $D_A$ and $D_B$ can form a social neighbourhood if $D_A$ and $D_B$ have an offline channel between them.
2) Let's assume that the offline channel between the device owners $A$ and $B$ has HTLCs with lock trees where each node has $k1$ children and there are $k2$ levels. In these settings, each device owner can distribute keys among $k1$ devices. Note that there are $k1$ sub-trees from the root node. All locks and key combinations in one such sub-tree will be assigned to one IoT device in the IoT network owned by $A$ or $B$.
3) Let the device $D_A$ gets the set of locks $(Lock_A^1, \ldots, Lock_A^{k3})$ and the device $D_B$ gets the set of locks $(Lock_B^1, \ldots, Lock_B^{k3})$.

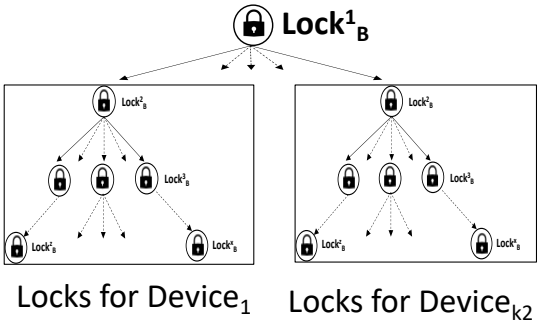


Fig. 5. Key distribution among the devices in an IoT network. Keys in each sub-tree can be assigned to one device. Each device can use a unique key every time they interact with another device from another IoT network.

4) $D_A$ and $D_B$ first exchange locks $Lock_A^i \in (Lock_A^1, \ldots, Lock_A^{k3})$ and $Lock_B^j \in (Lock_B^1, \ldots, Lock_B^{k3})$.
5) $D_A$ can verify ownership of $D_B$ by $B$ by checking the input transaction from $B$ to $M_{A,B}$ which will include the set of locks to be used by $B$ in the offline channel with $A$.
6) Similarly, $D_B$ can verify ownership of $D_A$ by $A$.
7) Next, $D_A$ will get $Key_A^i$ from $A$ and $D_B$ will get $Key_B^j$ from $B$. They will exchange these keys with existing key exchange protocols. $D_A$ will inform $Key_B^j$ to $A$ and $D_B$ will inform $Key_A^i$ to $B$.
8) $D_A$ will encrypt its message to $D_B$ with $Key_A^i$ and $D_B$ will encrypt its message to $D_A$ with $Key_B^j$.
9) $D_A$ will keep a timestamped message from $D_B$ which will include the following:
   a) Most recent blockchain head, i.e., Hash of the most recent block.
   b) The key $Key_B^j$.
   c) A new key $Key_B^x$ whose time-lock is not expired and whose key is not revealed yet.
   d) A digital signature of $B$ on the above message.
10) The encryption keys used by $D_A$ and $D_B$ will be updated after certain time intervals. The expiry time of an encryption key is denoted by the lock-time on the lock of the key in the HTLC contracts among $A$ and $B$. For example, $Key_B^j$ has a time lock 1 Day (measured in terms of the number of new blocks) then this key is valid for 1 day and it must be renewed before e day.
11) To renew the encryption key, $D_B$ produces the message mentioned in step 9 to $D_A$ and $D_A$ does the same.

### E. Extending social Neighbourhood

We will use path-based token transfer methods in blockchain offline channels to facilitate service provision by one device to another device who are not social neighbours (Shown in figure 7). Let the device $D_B$ wants to access the service provided by the device $D_A$. They are not social neighbours but there is another device $D_C$ who is a social neighbour of both $D_A$ and $D_B$. $D_B$'s can choose the service provided by $D_A$ if the following holds:

1) It can be proven that $D_A$ belongs to $A$.
2) It can be proven that $D_A$ and $D_B$ will communicate with keys that are not previously used.

The procedure used by $D_A$ and $D_B$ to exchange keys for data exchange and verifying the above properties is as follows:

1) $D_A$ informs $D_B$ about locks $H(Lock_A^1)$, a random lock $Random\ Lock_A$, and identity of the owner of $D_A$ as the public key of $A$. $D_A$ will pass this information to $A$. $D_B$ informs $D_A$ about locks $H(Lock_B^1)$, a random lock $Random\ Lock_B$, and identity of the owner of $D_A$ as the public key of $A$. $D_B$ will pass this information to $B$.
2) $A$ will find a path to $B$ in the channel network. Assume that the path is $A \rightarrow C$ and $C \rightarrow B$. A sequence of HTLCs will be created as follows:

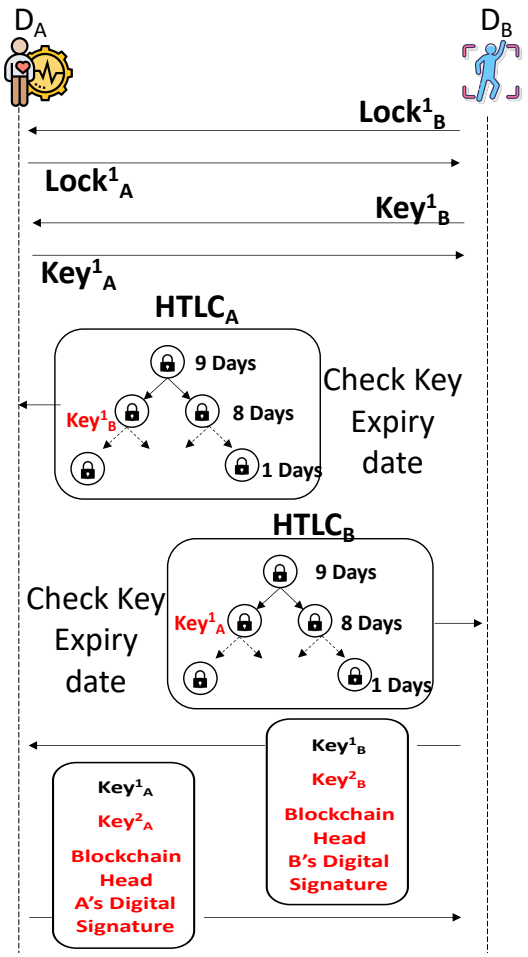


Fig. 6. Protocol used by two IoT devices to form a social neighbourhood between them. A device can check the validity of the encryption key to be used by another device by checking the blockchain if Hash of such key exists and if the key has not expired (denoted by time lock in the HTLC).

a) $HTLC_1$ created by $A$ and it states that from the multi-signature address $M_{A,C}$ 1 token will be given after 10 days if $C$ does not claim this token before 10 days by producing the key to $Random\ Lock_B$ and key to $H(Lock_B^1)$.
b) $HTLC_2$ created by $C$ and it states that from the multi-signature address $M_{C,B}$ 1 token will be given after 9 days if $B$ does not claim this token before 9 days by producing the key to $Random\ Lock_B$ and key to $H(Lock_B^1)$.

3) $B$ will find a path to $A$ in the channel network. Assume that the path is $B \rightarrow C$ and $C \rightarrow A$. A sequence of HTLCs will be created as follows:
a) $HTLC_3$ created by $B$ and it states that from the multi-signature address $M_{B,C}$ 1 token will be given after 10 days if $C$ does not claim this token before 10 days by producing the key to $Random\ Lock_A$ and key to $H(Lock_A^1)$.
b) $HTLC_4$ created by $C$ and it states that from the multi-signature address $M_{C,A}$ 1 token will be given after 9 days if $A$ does not claim this token before 9 days by producing the key to $Random\ Lock_A$ and key to $H(Lock_A^1)$.
4) $HTLC_1$ will be executed by $B$ as it will claim the tokens from $M_{C,B}$ by producing Keys to $Random\ Lock_B$ and $H(Lock_B^1)$. These keys will be used by $C$ to claim tokens from $HTLC_2$. After the execution $HTLC_2$, $A$ will know key to $Random\ Lock_B$ and $H(Key_B^1)$.
5) Similarly, $HTLC_3$ will be executed by $A$ and $HTLC_4$ will be executed by $C$. The result of these HTLC executions is $B$ will know the key to $Random\ Lock_A$ and $H(Key_A^1)$.
6) After execution of these HTLCs, $D_A$ and $D_B$ will exchange keys to $Lock_A^1$ and $Lock_B^1$. Previous steps verify that $D_A$ is owned by $A$ and $Lock_A^1$ is registered as a valid encryption key of devices of $A$. Next, $D_A$ and $D_B$ will use these keys to encrypt their messages as provide service to each other and send data can be securely exchanged between these devices.

The significance of the above protocol is as follows:

1) $A$ can verify that $B$ owns $D_B$ by checking the input transactions to $M_{B,C}$. Input transactions to this multi-signature address will contain lock $Key_B^1$ (as shown in section 3). Similarly $B$ can verify $A$ owns $D_A$.
2) One intermediator peer, in this example $C$, can not know the entire path from $A$ to $B$ if the path length is more than 2 channels. This means the intermediator nodes will not know that $D_B$ wants to use the service offered by $D_A$.
3) State of art fund routing protocols for blockchain offline channels such as [5], [6] can be used by $A$ and $B$ to find a path between them.

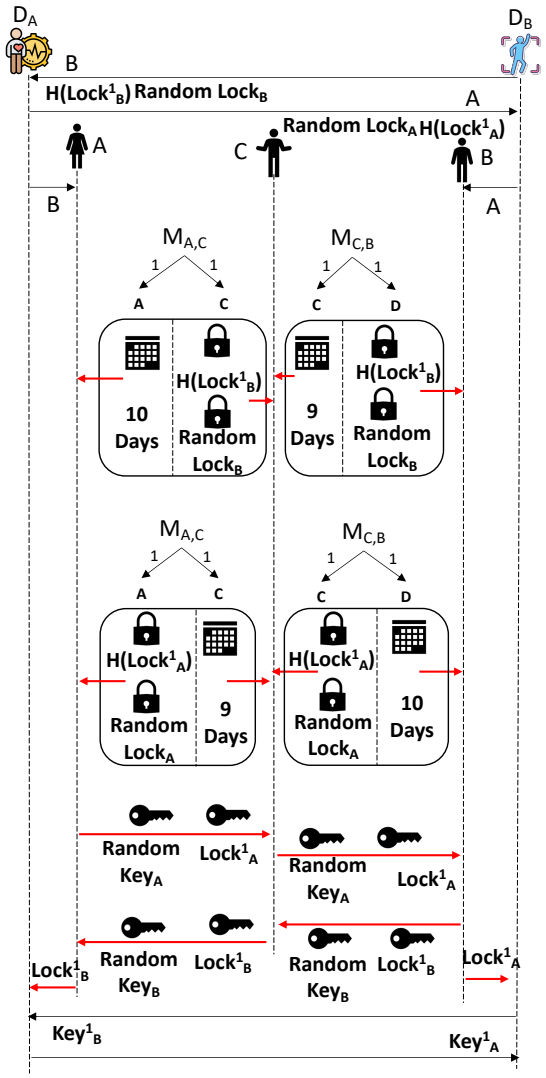


Fig. 7. Protocol used by one IoT device to find another IoT device and securely verify each other authenticity. This protocol allows devices to authenticate each other when their agents do not have channel between them.

4) If the protocol is successfully executed then it will mean there is a path in the SIoT network among the devices. The length of such a path may indicate the level of trust between two devices.

## IV. Analysis

**Theorem 1.** *The SIoT platform is privacy-preserving as only a device will know its social neighbours.*

*Proof.* Two IoT devices, $D_A$ from $A$ and $D_B$ from $B$ can form a social neighbourhood only using the encryption keys registered in the offline channel between $B$ and $A$. Hash of such encryption keys is included in the transactions corresponding to the offline channel between $B$ and $A$. Any peer in the blockchain network can see this transaction but they can not guess the encryption key as only its Hash is visible. A third

party can not see the social relation between $D_A$ and $D_B$ because:

1) If the third party is another device in the same IoT network of $D_A$, say $D'_A$ (also owned by $A$) then it will not know which device of $A$ got which sets of encryption keys as $A$ will only reveal a device's allocated set of encryption keys to it.
2) If the third party is another device owner say $C$, then it will not see the social relation between $D_A$ and $D_B$ as it can not guess the encryption key used by $D_A$ and $D_B$.

□

**Theorem 2.** *The SIoT platform is secure against the following adversaries:*

- *Similar to the replay attack on IoTs, an adversary may use an old encryption keys to access data from a device.*
- *Similar to the impersonation attack on IoTs, a malicious device controlled by an adversary may claim to be part of a specific IoT network and feed false data to another device.*
- *Devices owned by agents who are unknown to an agent $A$ or its neighbour in the social network may access the devices of $A$.*

*Proof.* The SIoT platform is secure because:

1) The old encryption key will not be reused. As shown in figure 6, a device will check the expiry of its key to be used for encrypting messages to another device. If a key is expired then it will not be possible to generate the time-stamped message indicating the time of exchange of the key. Note that, expiry of a key is indicated by the number of new blocks. If a message is created after these numbers of new blocks are created then another party can verify that the number of new blocks is more than what is labeled as the expiry date of the key. Thus expired encryption keys will not be used.
2) A device can verify the authenticity of another device (via its device owner) by checking if the Hash of the encryption key is included in the transactions funding the multi-signature address between its owner and another owner.
3) A device can verify the existence and length of the path which connects it to another device. As shown in figure 7, the protocol for finding another device can only work if there is a path between them.

□

**Theorem 3.** *The SIoT platform is highly scalable compared with generic blockchain-based IoT management solutions.*

*Proof.* It takes two transactions to open an offline channel and one transaction to close it. As shown in figure 4, an offline channel between two IoT device owners can allocate $k2$ encryption keys to $k1$ devices. Hashes of all these encryption keys are recorded in the blockchain in a transaction. In the worst case, we may need $k1 \times k2$ transactions (if one transaction stores one encryption key). In contrast, our proposed method will only use two transactions to allocate these encryption keys. This will significantly improve the scalability of the SIoT platform. □

## V. Conclusion

In this paper, we proposed a protocol to form decentralised SIoT network. We used blockchain offline channels in developing such a protocol. We proved that the protocol is secure as a device can not lie about its friendship with other devices. We will extend the outcomes of this paper for SIoT service discovery methods with location information on the devices. We will extend this decentralised SIoT network formation protocol to estimate trust among the devices.

## References


[1] L. Atzori, A. Iera, and G. Morabito, "Siot: Giving a social structure to the internet of things," *IEEE Communications Letters*, vol. 15, no. 11, pp. 1193–1195, 2011.

[2] L. Atzori, A. Iera, G. Morabito, and M. Nitti, "The social internet of things (siot) – when social networks meet the internet of things: Concept, architecture and network characterization," *Computer Networks*, vol. 56, no. 16, pp. 3594 – 3608, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128612002654

[3] M. Nitti, L. Atzori, and I. P. Cvijikj, "Network navigability in the social internet of things," in *2014 IEEE World Forum on Internet of Things (WF-IoT)*, 2014, pp. 405–410.

[4] P. Angin, M. B. Mert, O. Mete, A. Ramazanli, K. Sarica, and B. Gungoren, "A blockchain-based decentralized security architecture for iot," in *Internet of Things – ICIOT 2018*, D. Georgakopoulos and L.-J. Zhang, Eds. Cham: Springer International Publishing, 2018, pp. 3–18.

[5] S. Roos, P. Moreno-Sanchez, A. Kate, and I. Goldberg, "Settling payments fast and private: Efficient decentralized routing for path-based transactions," *CoRR*, vol. abs/1709.05748, 2017. [Online]. Available: http://arxiv.org/abs/1709.05748

[6] G. Malavolta, P. Moreno-Sanchez, A. Kate, and M. Maffei, "Silentwhispers: Enforcing security and privacy in decentralized credit networks," *IACR Cryptology ePrint Archive*, vol. 2016, p. 1054, 2016.

[7] A. F. Skarmeta, J. L. Hernández-Ramos, and M. V. Moreno, "A decentralized approach for security and privacy challenges in the internet of things," in *2014 IEEE World Forum on Internet of Things (WF-IoT)*, 2014, pp. 67–72.

[8] M. Nitti, L. Atzori, and I. P. Cvijikj, "Friendship selection in the social internet of things: Challenges and possible strategies," *IEEE Internet of Things Journal*, vol. 2, no. 3, pp. 240–247, 2015.

[9] J. Li, N. Zaman, and H. Li, "A decentralized locality-preserving context-aware service discovery framework for internet of things," in *2015 IEEE International Conference on Services Computing*, 2015, pp. 317–323.

[10] R. A. Rahman and B. Shah, "Security analysis of iot protocols: A focus in coap," in *2016 3rd MEC International Conference on Big Data and Smart City (ICBDSC)*, 2016, pp. 1–7.

[11] U. Jayasinghe, N. B. Truong, G. M. Lee, and T. Um, "Rpr: A trust computation model for social internet of things," in *2016 Intl IEEE Conferences on Ubiquitous Intelligence Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*, 2016, pp. 930–937.

[12] J. Li, Y. Bai, N. Zaman, and V. C. M. Leung, "A decentralized trustworthy context and qos-aware service discovery framework for the internet of things," *IEEE Access*, vol. 5, pp. 19 154–19 166, 2017.

[13] S. Guleng, C. Wu, X. Chen, X. Wang, T. Yoshinaga, and Y. Ji, "Decentralized trust evaluation in vehicular internet of things," *IEEE Access*, vol. 7, pp. 15 980–15 988, 2019.

[14] B. Pourghebleh, K. Wakil, and N. J. Navimipour, "A comprehensive study on the trust management techniques in the internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 9326–9337, 2019.

[15] J. Poon and T. Dryja, "The Bitcoin Lightning Network:Scalable Off-Chain Instant Payments." [Online]. Available: https://lightning.network/lightning-network-paper.pdf