Enabling the Sharing Economy: Privacy Respecting Contract based on Public Blockchain

Lei Xu, Nolan Shah, Lin Chen, Nour Diallo, Zhimin Gao, Yang Lu and Weidong Shi
Computer Science Department, University of Houston
Houston, Texas, United States
xuleimath@gmail.com, nolanshah212@gmail.com, chenlin198662@gmail.com, noudiallo@gmail.com, mtion@hotmail.com, ylu17@uh.edu, wshi3@uh.edu

ABSTRACT

Blockchain is a novel way to construct fully distributed systems and has the potential to disrupt many businesses including Uber and Airbnb within the sharing economy. Specifically, blockchain provides a method to enforce the agreement between a user and the physical property owner without using any trusted party, e.g., if a user pays the agreed money, the blockchain guarantees that he/she has access to the property. While a blockchain based system has many desirable features, it may leak privacy information of involved parties due to its openness to the public. To mitigate the privacy concern, we propose a privacy respecting approach for blockchain-based sharing economy applications, which leverages a zero-knowledge scheme. We also analyze the security features and performance of the proposed approach to demonstrate its effectiveness in these applications.

CCS Concepts

•Security and privacy → *Access control;* **•Computer systems organization** → *Embedded and cyber-physical systems;*

Keywords

IoT; sharing economy; public blockchain; privacy

1. INTRODUCTION

The emerging blockchain technology offers a novel way for bookkeeping in a fully distributed manner. Initially, it was only used to build e-cash systems [28], but later it's potential to innovate other areas where there is a lack of trust between involved parties was realized [36]. One such area is the sharing economy. For example, the German company Slock.it has developed the Ethereum Computer which joins smart objects (such as locks, washing machines, or vehicles) with the blockchain so these objects are rentable with little trust or interaction between owner and renter [35]. The Ethereum Computer allows the owner of a smart object to create a smart contract on the Ethereum Blockchain that governs transactions and access control for that object. After a renter finishes paying the owner the listed amount to use the object, he/she will be granted access for an allotted time and can manipulate the object's state. This approach has many advantages compared with existing centralized approaches, including lower cost (users do not need to pay the centralized broker) and higher reliability (there is no single point of failure, and the storage is immutable). Therefore, blockchain applications like Slock.it are believed to be able to disrupt the disruptors like Airbnb and Uber [39, 21].

One major issue to replacing centralized systems in sharing economy applications is privacy, i.e., in order to enforce a contract with blockchain, identities of parties involved in the contract has to be disclosed together with the contract. For example, a renter does not want to show his/her itinerary to the public by disclosing contracts with different property owners. It is relatively easy to protect privacy sensitive information with a private or permissioned blockchain [19]. However, these blockchain systems are more like variants of classical distributed system and do not support full decentralization. Public blockchains are a better choice because they are fully open systems and do not consider hiding information from participants.

To address the privacy concern without sacrificing the desirable features of public blockchain, we propose a privacy respecting contract platform based on the public blockchain, PrC, that can be used to build a variety of sharing economy applications. The proposed scheme does not assume any special party in the blockchain system and does not change the overall design of existing applications. It leverages a noninteractive zero-knowledge technique and can be applied to many other IoT areas. In summary, our contributions in this paper include:

- We provide detailed design of PrC that offers a privacy respecting contract platform;
- PrC adds more privacy features to public blockchain and addresses the major concern that hinders the sharing economy applications to migrate to the public blockchain environment; and
- We analyze the security features of PrC and evaluate its performance to show its practicality.

The remainder of the paper is organized as follows: In Section 2 we give a short review of the blockchain technology and cryptography tools used in PrC. Section 3 provides a detailed description of PrC, and Section 4 evaluates performance and security features of the proposed approach.

BCC'17, April 2, 2017, Abu Dhabi, United Arab Emirates.
© 2017 Copyright held by the owner/author(s). Publication rights licensed to ACM. ISBN 978-1-4503-4974-1/17/04...$15.00.
DOI: http://dx.doi.org/10.1145/3055518.3055527

Section 5 discusses related work, and we conclude the paper in Section 6.

2. BACKGROUND OF TECHNOLOGIES

In this section, we briefly review background technologies.

2.1 Blockchain

Blockchain is at the core of the cryptocurrency, Bitcoin [28], and a major technology contributor to the success of Bitcoin over a community of distributed peers. It is believed that blockchain based technology may revolutionize many industries [37, 36].

Roughly speaking, a blockchain is a system that involves multiple participants who achieve consensus over a dataset and maintain the data locally. Blockchain systems are developed under different trust models with different consensus protocols. There are two main types of trust models: one assumes all participants are equivalent (public blockchain) and one has participants with different privileges for block construction (private/permissioned/federated blockchain). Under a given trust model, the system can use various consensus protocols including proof-of-work [28], proof-of-stake [6], or BFT [38]

Regardless of the trust model and underlying consensus protocol, most blockchain systems have three important features [31, 20]: (i) public accessibility (all information stored with block-chain is publicly accessible to everyone); (ii) immutability (information added to the blockchain is not modifiable or removable); and (iii) resilience (each participant of the system keeps a complete copy of the blockchain, and no single point of failure can affect the availability of the stored information).

There are four major operations for blockchain: (i) checking validity of received record; (ii) generating a block to hold records and verifying validity of received blocks; (iii) checking the validity of a record/block; and (iv) producing a new record according to instructions of existing records. In our case, the last operation is used to facilitate the enforcement of contracts.

2.2 Cryptographic Tools

Two major cryptographic tools are used in this work, commitment scheme, and non-interactive zero-knowledge proof.

Commitment scheme. A commitment scheme [5, 10, 12] involves two players, the *committer* and the *receiver*, running probabilistic polynomial-time algorithms. The execution of the scheme is divided into two phases. During the first phase, the *committer* chooses a value v, and runs `COMMIT` to generate a commitment on v, which is shared with the *receiver*. During the second phase, the *committer* shares extra information with the *receiver*, and the *receiver* can execute `REVEAL` to verify the committed value v.

A secure commitment scheme has two major features: (i) Hiding. The *receiver* cannot learn v from the commitment; (ii) Binding. The *committer* cannot change v to v' after the first phase.

Zero-knowledge proofs. The zero-knowledge proof is also an important cryptographic primitive [16]. Roughly speaking, a zero-knowledge proof involves two parties, the *prover* and the *verifier*. For a statement, the *prover* can generate a proof to convince the *verifier* the correctness of the statement. In this process, the *verifier* cannot learn anything except the fact that the statement is true (zero-knowledge feature). Zero-knowledge proofs can be either interactive or non-interactive. Interactive zero-knowledge proofs [17, 15] requires the *prover* to communicate with the *verifier* multiple rounds to finish the proof. Non-interactive zero-knowledge proofs (NIZK) [4, 18, 9] do not require multiple rounds interaction between *prover* and *verifier* and is more suitable for scenarios where it is hard for these parties to be online at the same time.

An important tool that related to NIZK is zero-knowledge succinct non-interactive arguments of knowledge (zk-SNARK) [14, 2, 3]. zk-SNARK allows for efficient verification of NP statements with proofs produced by an untrusted but computationally bounded prover. A trusted party is used to publish a proving key and verification key, where the proving key is used to generate non-interactive proofs for adaptively-chosen NP statements, and the verification key is used to verify the proof. We refer the readers to [2] for a more formal definition of zk-SNARK.

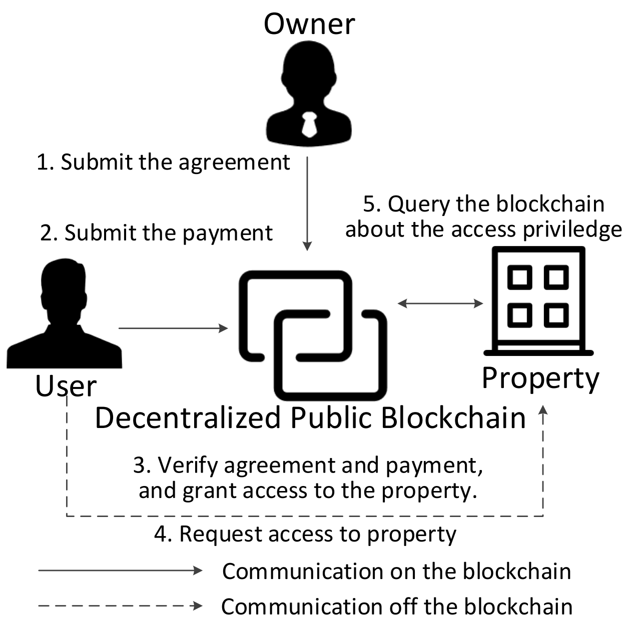
Owner
1. Submit the agreement
5. Query the blockchain about the access priviledge
2. Submit the payment
User
Property
Decentralized Public Blockchain
3. Verify agreement and payment, and grant access to the property.
4. Request access to property
Communication on the blockchain
Communication off the blockchain

Figure 1: Overview of the workflow of using blockchain for room sharing. The privacy protection mechanism is not reflected in the figure and discussed later.

3. BLOCKCHAIN BASED PHYSICAL ACCESS

In this section, we provide a detailed description of PrC. PrC can be deployed for various blockchain based sharing economy applications that involved access control of physical properties and has the potential to extend to a broader range of other online-to-offline applications and these related to IoT. For convenience, we use the case of room sharing to illustrate the proposed approach.

3.1 Security Model and Overview of the System

The overall workflow of the proposed scheme is depicted in Fig. 1. PrC for blockchain-based room sharing involves four parties:

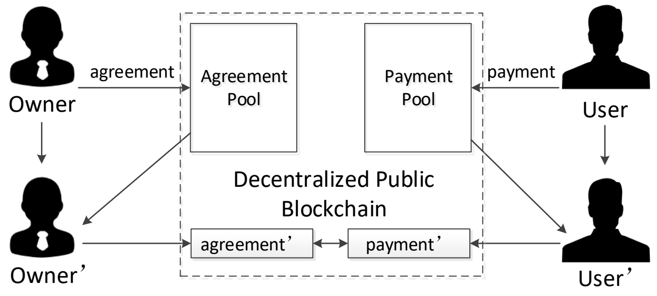

Figure 2: The basic idea of PrC to protect privacy information of the owner and user. Instead of doing the transaction directly, the owner and user use proxies to finish the renting and access granting procedure.

- Owner. The property owner rents his/her rooms to the user in exchange for money. The property owner is not trusted, and he/she may try to maximize his/her benefit by different approaches including to concurrently rent the property to different users;
- User. The user rents a room from the owner through the blockchain. He/she is not fully trusted. He/she may try to minimize costs by avoiding payment to the owner. Both the owner and the user have the incentive to protect their privacy;
- Physical property. The physical property is rented to the user by the owner. We assume the physical property is controlled by machines and pre-determined protocols and therefore is trusted, i.e., the property is programmed to follow decisions made by the blockchain system. It is also assumed that the physical property will not disclose information to any third party; and
- Blockchain. The blockchain is the platform to carry out transactions between owners and users by recording agreements/payments and tracking agreement execution. The blockchain is accessible to anyone including attackers. It is maintained by a large number of participants through mining, where the majority of participants are honest. Trusted properties always follow instructions accepted by the blockchain system.

If the owner and user submit transaction information directly to the blockchain in plain-text, anyone can observe the blockchain and learn their activities. The goal of PAC is *to protect the privacy of both the owner and the user, i.e., an adversary with access to the blockchain cannot learn information like who rents the owner's room and how much the user pays for the room.*

Fig. 2 shows the idea behind PrC to protect privacy. The renting agreement (generated by the owner) and payment (generated by the user) do not link with each other directly on the blockchain. Instead, they are submitted to corresponding pools first. Two temporary agents are used to pull out corresponding records and commit to the blockchain again in a new form. Although the new agreement and payment are connected on the blockchain, the adversary cannot learn the relationship with original ones, and privacy is preserved.

3.2 Basic Operations of PrC

We first consider the case where the time and price are related, e.g., each time the user pays a fixed price to the owner for a fixed time frame, and discuss the case where time and price are unrelated after.

Initialization. At this stage, public parameters used in following steps are generated and released to the public. Because there is no secret, anyone can be in charge of the generation process and this does not affect security assumptions of PrC. Public parameters include commitment scheme and zk-SNARK scheme used in the system, and common parameters of these schemes.

In this step, the potential owner and user also exchange information about agents they plan to use for the transaction. The exchange can be done through an off-blockchain channel, or use key-private encryption scheme [1] to encrypt the message with receiver's public key and store to the blockchain. The key-private feature guarantees that an adversary observing cipher-texts stored on the blockchain cannot infer the identity of the receiver.

Owner creates an agreement. The owner creates the agreement in two steps:

1. The owner O generates a transaction

   $$\mathrm{tx}_{\mathrm{rsv}} = (t, cm_t)$$

   and submits it to the blockchain. t is the time period that the owner O reserved for a renter, and $cm_t = \texttt{COMMIT}(sn_t, r_t)$ is the commitment of sn_t, which is a piece of information that O generated for the reservation of the room at time t. One straightforward way to construct sn_t is to use the public key of the agent of the user (which is determined in the initialization phase) to encrypt the time information. r_t is the random number that O generates for the commitment cm_t.

   Participants of the blockchain check whether t is valid, i.e., the owner O does not use t before. If the transaction is valid, $\mathrm{tx}_{\mathrm{rsv}}$ is accepted by the blockchain. The hiding property of the commitment scheme guarantees that an adversary cannot learn information about sn_t by observing cm_t.

2. The owner O constructs a zk-SNARK proof π_{agmt} on the NP statement "I know r_t such that $\texttt{COMMIT}(sn_t, r_t)$ is on the blockchain". Then, O generates a proxy O', and O' submits the following transaction to the blockchain:

   ```
   $\mathrm{tx}_{\mathrm{agmt}} = ($
   $\pi_{\mathrm{agmt}}, sn_t, c_{r_t}$
   if Received payment for $O'$ then
       Post $\mathrm{tx}_{\mathrm{grant}} = (U', sn_t)$ to the blockchain
   end if
   $),$
   ```

   where c_{r_t} is the cipher-text of r_t, which is encrypted with public key of the agent of the targeted user. With π_{agmt}, participants of the blockchain can verify the validity of sn. They also check whether sn has been posted on the blockchain before. This prevents double-usage, i.e., the owner cannot rent the room to more than one user at the same time. If $\mathrm{tx}_{\mathrm{agmt}}$ passes all checks, it is accepted by the blockchain as a new record. Note that one cannot post tx_{grant} directly without payment as the system can detect and reject such request.

The relationship between O and O' is not disclosed as an adversary cannot establish a connection between tx_{agmt} and tx_{rsv} by extracting information from the zk-SNARK proof and commitments on the blockchain.

User makes a payment. After the owner finishes these two steps, he/she notifies the user U. The user U checks the blockchain to verify the agreement tx_{agmt}, including verifying that O' owns sn_t and the relationship between sn_t with t (by decrypting sn_t and c_{r_t} to verify the commitment). Because sn_t has been published on the blockchain, U cannot claim it again even if he learns r_t.

and then starts the payment process, which divides into two steps:

1. The user U generates a transaction

$$\text{tx}_{\text{pre}} = (v, cm_v)$$

and submits to the blockchain. v is the value of the payment and $cm_v = \texttt{COMMIT}(sn_v, r_v)$ is the commitment of sn_v which is a random serial number that U generated for the payment. r_v is the random number used to build the commitment.

Before accepting tx_{pre} as a new record on the blockchain, participants deducts v from the account of user U. If U does not have enough balance, tx_{pre} is rejected.

2. The user U constructs a zk-SNARK proof π_{pay} on the NP statement "I know r_v such that $\texttt{COMMIT}(sn_v, r_v)$ is on the blockchain". Then, user U generates a proxy U', and U' submits the following transaction to the blockchain:
$\text{tx}_{\text{pay}} = ($
π_{pay}
Transfer sn_v to O'
Trigger tx_{agmt}
$)$

The blockchain verifies π_{pay}, transfers the money to O', and revisit tx_{agmt}. As the condition given in tx_{agmt} is satisfied after the payment, the system works to post the record $\text{tx}_{\text{grant}} = (U', sn_t)$ to the blockchain.

When the user finishes the payment, O' can transfer the money to O. As the price is fixed, an adversary cannot establish the relationship between a payment and a room sharing contract.

User accesses the property. The user uses identity U' to request access to the property, and the identity information of the property is also recorded on the blockchain (denoted as P). To prevent a man-in-the-middle attack, U' and P can run a mutual authentication to verify each other. P checks the blockchain for the access privilege of U'. This operation does not generate any new records on the blockchain and P follows related records to grant or deny the access request of U'. Specifically, as P is assumed to be a trusted party, U' can disclose his/her private key to P, and P can decrypt sn_t to learn the time period that U' is allowed to enter.

3.3 Other Operations and Refinements

Cancellation operations. Besides the basic operations like generating agreement and making payment, another useful operation that PrC supports is *cancellation*, which allows the owner and user to change their minds at certain stages of the transaction.

- If cancellation occurs between the first and second step of agreement creation, the owner O can disclose both sn_t and r_t to the blockchain and request to revert the effect of tx_{rsv}. The blockchain verifies whether these values are consistent and generates a new record to add time t back to the owner O's account on available time;
- If cancellation occurs between agreement creation (owner's second step) and user payment (user's first step), then the owner O and his/her proxy agent O' can generate a record together to disclose r_t and request the system to revert. The blockchain can verify whether r_t is valid (by checking the commitment value), and generate a new record to return the time t to O directly. In this case, the relationship between O and O' is disclosed. This is not a problem because O only uses O' once, and he/she generate another O'' for next usage.
- If cancellation between the first and second step of user payment, then the user U can disclose sn_v and r_v to request the refund. The blockchain checks the commitment and generates a new record to refund the money to U.
- If cancellation occurs after the user payment, then the user (U and U') has to work together with the owner (O and O') to cancel the transaction. They have to jointly construct a record to disclose both r_t and r_v, and request to cancel the transaction. If the user U does not have enough balance on his/her account, the cancellation fails. Otherwise, the blockchain verifies r_t and r_v with the commitment values to add time/money back to the owner/user. Cancellation at this stage requires the disclosure of the relationship between the user and the owner, and privacy is not preserved.

PrC does not monitor any offline activities. If the user requests to cancel the transaction after he/she has used the property, it is the responsibility of the owner to refuse the cancellation request.

Efficient data organization. PrC involves verification operations of zk-SNARK proof. If reservation records (tx_{rsv} and pre-payment records (tx_{pre}) are stored on the blockchain in a linear way, cost of verification of the proof increases quickly. Instead, all these records can be organized as leaves of a Merkel tree [26], where the verification cost only increases logarithmically to the number of records. Specifically, we fix a large enough value for the height of the tree (e.g., with height 32, the tree can handle 2^{32} records), and all records are added as leaf nodes. Internal nodes are updated accordingly and the value of the root is included in the zk-SNARK proof [3]. This does not affect any security feature of the zk-SNARK scheme or the overall system.

Supporting different time lengths and prices. The approach described in Section 3.2 has the limitation that for one contract, the time and price are fixed. The owner and the user may need to have multiple contracts for a long stay or high price.

First we consider changing the time length for the owner to create the agreement. Instead of including t in tx)rsv, the owner puts a time range (t_1, t_2) into the record. As the information is in plain-text, the blockchain can still verify whether that time period is available. sn_t is also modified accordingly to incorporate this information to allow the potential user to verify before making a payment. Allowing the user to pay different amount of currency for the property is more complex as the payment value may leak connection between the owner and the user. We can leverage the scheme used by Zerocash to divide the payment to O and O' in a random way to hide the connection [33].

4. PERFORMANCE EVALUATION OF DL-BAC

In this section, we analyze security features and performance of PrC.

4.1 Security Analysis

Double usage. Double usage has two aspects for the room sharing application (and other similar sharing economy applications). The owner tries to rent the room to multiple users, and the user uses the same payment for different rooms. If both the owner and the user follow the two-steps procedures, PrC can always reject any records that try to use the same resource twice. If the owner tries to ignore the first step of agreement creation, he/she cannot generate a valid zk-SNARK proof and cheat the blockchain to accept a record of the second step. This prevents the owner from having a contract with the user without available property. The situation is the same for users.

The owner cannot create multiple contracts with overlapped time for different users neither. Specifically, before a user makes payment, he can verify whether the sn_t is valid by tracing back to the record that creates sn_t.

Owner and user privacy. Protecting owner and user privacy is the goal of the PrC. By observing records stored on the blockchain, an adversary should not determine whether an owner and a user have had a contract in the past. Although the connection between proxy agents is disclosed on the blockchain, it is hard for an attacker to infer identity information related to the agents. The only connection between the original identity and proxy agent identity is a zk-SNARK proof. The zero-knowledge feature prevents any attacker with limited computation capability (probabilistic polynomial time algorithm) to recover the connection between the two identities.

4.2 Performance Analysis

Performance of PrC is affected by two factors: performance of the underlying blockchain and performance of involved cryptographic primitives.

Performance of Blockchain. PrC is not bound with any specific blockchain system. The original Bitcoin protocol can only support limited throughput [13] and may not be able to satisfy the demands of various sharing economy applications developed on top of PrC. A lot of work has been done to improve the performance of blockchain system and

Table 1: Performance of existing zk-SNARK schemes [3, 30].

Operation	Time
Key generation	117 s to 123 s
Proof generation	147 s to 784 s
Verification	< 10 ms
Proof size	288 Bytes

PrC can take advantage of these works to improve the performance [11, 8, 25].

Performance of cryptographic primitives. Two major primitives involved in PrC are the commitment scheme and the zk-SNARK scheme. The commitment scheme is relatively cheap, so we focus on the extra cost related to zk-SNARK scheme. Key performance parameters of a zk-SNARK scheme includes key generation, proof generation, and verification. For existing zk-SNARK schemes, key generation is more expensive than proof generation, and proof generation is more expensive than verification. For a security level of 128 bits, the running times of these operations for a 1-million-gate circuit and 1000-bit input on a common computer are summarized in Table 1.

Key generation only needs to be executed once, so the related cost is not an issue. Proof generation is not cheap, but it is only run by the owner and user, and not the whole blockchain system. Therefore, it does not significantly affect the overall performance of the system. Verification is run by every participant of the blockchain, but the cost is cheap. Furthermore, proofs that have to be stored on the blockchain are very compact, so it is not a problem for every participant of the blockchain to keep a copy of these proofs.

5. REMARKS AND RELATED WORKS

In this section, we shortly review related works.

5.1 Remarks and Takeaways

In general, it is a hard problem to conceal identities involved in a contract with public blockchain, especially when the contract is related to the real physical world (e.g., contracts for room sharing). The approach proposed in this work uses proxy agents to execute the contract and leverages cryptography tools to cut off the connection between the real identities and their proxy agents. Although this two-layer approach can support most of the common operations that are necessary for sharing economy applications, it is not clear at this moment how to support ratings and recommendations. Future directions of research include providing a richer set of operations and evaluating the capability to recover connections between identities using other sources of information.

5.2 Related Works

IoT and blockchain. Blockchain technology has been used in different ways in IoT systems. Hardjono and Smith proposed the use of a permissioned blockchain for IoT device commissioning and data sharing [19]. Christidis and Devetsikiotis proposed a design of using blockchain platform for smart contracts and IoT [7]. [23] proposed to use blockchain to facilitate firmware updating of IoT devices. These works explored new opportunities of blockchain in the area of IoT but did not consider privacy protection problems.

Privacy in access control. Access control is an essential part of many sharing economy applications including room sharing. In scenarios where there is a centralized party, various techniques are proposed to achieve privacy protection in user authentication and access control [32, 29, 34, 24]. These approaches do not apply to the blockchain scenario, especially public blockchain where there are no nodes with special privilege.

Privacy in blockchain based transactions. Classical blockchain based transaction systems like Bitcoin [28] allow a user to hide his relationship with identities used in the system. More sophisticated systems like Zerocash [33] and Zerocoin [27] allow users involved in a transaction to hide their relationship with each other. These works focus on one-way currency transaction, but for sharing economy applications, the interaction is usually two-way and involves extra features beyond the currency. [22] proposed a general approach to executing smart contracts securely on the blockchain. This is orthogonal to our work as we focus on the privacy of the identities involved.

6. CONCLUSION

As an attracting platform of sharing economy applications, public blockchain lacks the capability of privacy protection as it is open to the public. This paper proposes PrC, a privacy respecting blockchain based sharing economy platform. PrC uniquely integrates cryptography tools and blockchain technology, and achieves the goal of maintaining desirable features that public blockchain offered to sharing economy applications without sacrificing user's privacy. Although we use the case of room sharing to describe the approach, PrC can be extended to other areas with similar setting and opens the door to exploring more sharing economy applications where user identity is privacy sensitive.

For the next step, we plan to implement a fully functional prototype of PrC and evaluate it with different system sizes and applications. We will also extend PrC to support other functionalities that are useful for sharing economy like privacy preserving rating and recommendation.

7. ACKNOWLEDGEMENT

The authors would like to thank the anonymous reviewers for their valuable comments.

8. REFERENCES

[1] M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-privacy in public-key encryption. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 566–582. Springer, 2001.

[2] E. Ben-Sasson, A. Chiesa, D. Genkin, E. Tromer, and M. Virza. Snarks for c: Verifying program executions succinctly and in zero knowledge. In *Advances in Cryptology–CRYPTO 2013*, pages 90–108. Springer, 2013.

[3] E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza. Succinct non-interactive zero knowledge for a von neumann architecture. In *23rd USENIX Security Symposium (USENIX Security 14)*, pages 781–796, 2014.

[4] M. Blum, P. Feldman, and S. Micali. Non-interactive zero-knowledge and its applications. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 103–112. ACM, 1988.

[5] G. Brassard, D. Chaum, and C. Crépeau. Minimum disclosure proofs of knowledge. *Journal of Computer and System Sciences*, 37(2):156–189, 1988.

[6] V. Buterin. What proof of stake is and why it matters. *Bitcoin Magazine, August*, 26, 2013.

[7] K. Christidis and M. Devetsikiotis. Blockchains and smart contracts for the internet of things. *IEEE Access*, 2016.

[8] N. T. Courtois, P. Emirdag, and D. A. Nagy. Could bitcoin transactions be 100x faster? In *Security and Cryptography (SECRYPT), 2014 11th International Conference on*, pages 1–6. IEEE, 2014.

[9] A. De Santis, G. Di Crescenzo, R. Ostrovsky, G. Persiano, and A. Sahai. Robust non-interactive zero knowledge. In *Annual International Cryptology Conference*, pages 566–598. Springer, 2001.

[10] G. Di Crescenzo, J. Katz, R. Ostrovsky, and A. Smith. Efficient and non-interactive non-malleable commitment. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 40–59. Springer, 2001.

[11] I. Eyal, A. E. Gencer, E. G. Sirer, and R. Van Renesse. Bitcoin-ng: A scalable blockchain protocol. In *13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16)*, pages 45–59, 2016.

[12] M. Fischlin and R. Fischlin. Efficient non-malleable commitment schemes. In *Annual International Cryptology Conference*, pages 413–431. Springer, 2000.

[13] J. Garay, A. Kiayias, and N. Leonardos. The bitcoin backbone protocol: Analysis and applications. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 281–310. Springer, 2015.

[14] R. Gennaro, C. Gentry, B. Parno, and M. Raykova. Quadratic span programs and succinct nizks without pcps. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 626–645. Springer, 2013.

[15] O. Goldreich and A. Kahan. How to construct constant-round zero-knowledge proof systems for np. *Journal of Cryptology*, 9(3):167–189, 1996.

[16] O. Goldreich, S. Micali, and A. Wigderson. Proofs that yield nothing but their validity or all languages in np have zero-knowledge proof systems. *Journal of the ACM (JACM)*, 38(3):690–728, 1991.

[17] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof-systems. In *Proceedings of the 7th annual ACM Symposium on Theory of Computing - STOC 1985*, pages 291–304. ACM, 1985.

[18] J. Groth, R. Ostrovsky, and A. Sahai. Perfect non-interactive zero knowledge for np. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 339–358. Springer, 2006.

[19] T. Hardjono and N. Smith. Cloud-based commissioning of constrained devices using permissioned blockchains. In *Proceedings of the 2nd*

ACM International Workshop on IoT Privacy, Trust, and Security, pages 29–36. ACM, 2016.

[20] N. Hayase. The Blockchain and the Rise of Networked Trust. http://www.coindesk.com/blockchain-rise-networked-trust/, 2014.

[21] S. Huckle, R. Bhattacharya, M. White, and N. Beloff. Internet of things, blockchain and shared economy applications. *Procedia Computer Science*, 98:461–466, 2016.

[22] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. *University of Maryland and Cornell University*, 2015.

[23] B. Lee and J.-H. Lee. Blockchain-based secure firmware update for embedded devices in an internet of things environment. *The Journal of Supercomputing*, pages 1–16, 2016.

[24] A. Y. Lindell. Anonymous authentication. *Journal of Privacy and Confidentiality*, 2(2):4, 2007.

[25] L. Luu, V. Narayanan, K. Baweja, C. Zheng, S. Gilbert, and P. Saxena. Scp: a computationally-scalable byzantine consensus protocol for blockchains. Technical report, Cryptology ePrint Archive, Report 2015/1168, 2015.

[26] R. C. Merkle. A digital signature based on a conventional encryption function. In *Conference on the Theory and Application of Cryptographic Techniques*, pages 369–378. Springer, 1987.

[27] I. Miers, C. Garman, M. Green, and A. D. Rubin. Zerocoin: Anonymous distributed e-cash from bitcoin. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 397–411. IEEE, 2013.

[28] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008.

[29] L. Nguyen and R. Safavi-Naini. Dynamic k-times anonymous authentication. In J. Ioannidis, A. Keromytis, and M. Yung, editors, *Applied Cryptography and Network Security - ACNS 2005*, volume 3531 of *LNCS*, pages 318–333. Springer, 2005.

[30] B. Parno, J. Howell, C. Gentry, and M. Raykova. Pinocchio: Nearly practical verifiable computation. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 238–252. IEEE, 2013.

[31] M. Pilkington. Blockchain technology: principles and applications. *Research Handbook on Digital Transformations, edited by F. Xavier Olleros and Majlinda Zhegu. Edward Elgar*, 2016.

[32] S. Ruj, M. Stojmenovic, and A. Nayak. Privacy preserving access control with authentication for securing data in clouds. In *Cluster, Cloud and Grid Computing (CCGrid), 2012 12th IEEE/ACM International Symposium on*, pages 556–563. IEEE, 2012.

[33] E. B. Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *2014 IEEE Symposium on Security and Privacy*, pages 459–474. IEEE, 2014.

[34] S. Schechter, T. Parnell, and A. Hartemink. Anonymous authentication of membership in dynamic groups. In *International Conference on Financial Cryptography*, pages 184–195. Springer, 1999.

[35] Slock.it. Slock.it faq, 2016.

[36] M. Swan. *Blockchain: Blueprint for a new economy.* " O'Reilly Media, Inc.", 2015.

[37] S. Underwood. Blockchain beyond bitcoin. *Communications of the ACM*, 59(11):15–17, 2016.

[38] M. Vukolić. The quest for scalable blockchain fabric: Proof-of-work vs. bft replication. In *International Workshop on Open Problems in Network Security*, pages 112–125. Springer, 2015.

[39] G. Zervas, D. Proserpio, and J. Byers. The rise of the sharing economy: Estimating the impact of airbnb on the hotel industry. *Boston U. School of Management Research Paper*, (2013-16), 2016.