2018 IEEE Confs on Internet of Things, Green Computing and Communications, Cyber, Physical and Social Computing, Smart Data, Blockchain, Computer and Information Technology, Congress on Cybermatics

IoT Meets Blockchain: Parallel Distributed Architecture for Data Storage and Sharing

Shaowei Liu, Jing Wu, Chengnian Long*
Department of Automation, Shanghai Jiao Tong University, and the
Key Laboratory of System Control and Information Processing,
Ministry of Education of China, Shanghai 200240, China
Email: longcn@sjtu.edu.cn

***Abstract*—The rapid development of Internet of Things technology has led to the sharp increase of the number of interconnected devices. A large amount of data is generated by large-scale IoT devices. In this paper, we propose PDash with parallel distributed architecture, a scalable blockchain-based platform for IoT data storage and sharing. PDash aims to solve the bottleneck problem of blockchain system: scalability, and makes it possible for users to keep ownership of their own data as well as share data securely with economic feedback. PDash integrates decentralized blockchain network and distributed storage network to achieve secure data storage, validation and sharing, and utilizes multiple encryption technologies to enable efficient interaction between two parallel distributed networks.**

I. Introduction

Blockchain technology has exhibited considerable capacity in solving data security and privacy issues [1]. Blockchain is an open and decentralized database maintained by all nodes that participate in the network. Establishing a public fundamental data platform using blockchain can efficiently reduce connectivity cost of devices and make it possible that all data are analyzed and processed synthetically. Data centers are not necessary in blockchain-based IoT systems. The disappearance of data centers reduces the risk of data leakage from the attack. Users control their own data and safeguard their rights and interests. Blockchain entirely changes the roles of users, devices and manufacturers. It has the full potential of establishing a new infrastructure of internet of things. However, current blockchain systems have a fatal defect: scalability. To ensure stable operation of blockchain system, all nodes have to reach a consensus on every instruction, resulting in high cost of data storage and computing.

The scalability issue of blockchain systems consist of two levels, data and transactions. The existing blockchain systems treat data as part of the transaction and package it into the block, confirming the transaction by proof-of-work, a consensus mechanism that consuming a great deal of hash power [2]. In view of the scalability problem of blockchain, current efforts mostly focus on the transaction level. We divide these works into four directions. The first type is to design new consensus algorithms with higher efficiency. Such works can be divided into two categories [3] further. One is to replace the hash calculation in the proof-of-work with other faster methods such as proof-of-stake [4] and proof-of-activity [5], and the other is to design a hybrid consensus protocol that applies traditional Byzantine fault-tolerant algorithm to large-scale blockchain systems through the election of committee. For example, Algorand [6] designed mechanisms based on Verifiable Random Functions to decide whether nodes have been selected to the committee in which chosen nodes could thus be able to participate in the BA* protocol [7] to agree on the next block of transactions; Pass and Shi proposed to combine the inefficient proof-of-work protocol with a fast permissioned byzantine consensus protocol called DailyBFT to achieve a responsive permissionless consensus [8]. The second type is the side chain. Lightning network [9] is one of the most representative examples. The initial definition of side chain is that it is a protocol that allows bitcoin to transfer securely from main blockchain to another chain that runs in parallel to main blockchain, and return to main blockchain. The lightning network is a side chain for bitcoin blockchain. It allows two parties of the transaction to put bitcoin into a payment channel that locked by multi-signatures, then they can change the amount of bitcoins that each can retrieve. Either party can close the channel and the latest state will be signed and broadcast to the network. The last transaction will be written to main blockchain and bitcoin will return to main blockchain. The third type is called sharding. Sharding is a traditional database technology that divides huge database into smaller and faster parts, these parts are called shards. Applying sharding technology into blockchain is to divide blockchain network into smaller component networks that are able to handle transactions to achieve faster transaction rate [10]. The last type is the directed acyclic graph (DAG). The most famous DAG scheme is the Tangle proposed by IOTA [11]. DAG is a different data structure from blockchain. It organizes all transactions in the form of directed acyclic graph and is also decentralized. To issue a new transaction, Users have to choose another two transactions and verify them with lightweight POW computation, then link the new transaction to them and broadcast to the network.

There are also several works that try to address the scalability problem of blockchain at the data level. For

*Corresponding author.

978-1-5386-7975-3/18/$31.00 ©2018 IEEE
DOI 10.1109/Cybermatics_2018.2018.00233

example, IPFS [12] combines four advanced technologies of DHT, BitTorrent, Git and SFS to build a new generation Web protocol. IPFS can be used as the underlying storage platform of blockchain, but it is originally a distributed file system, there is no built-in optimization and data encryption for the blockchain system. In order to realize the interaction between data and blockchain, two major challenges of encryption and secure data sharing must be solved. In addition, there have recently been some blockchain-based decentralized storage systems such as Storj [13] and Filecoin [14]. They implemented an economically driven approach to file storage systems, but they did not address the issue of safe sharing across a large number of IoT data either. On the basis of distributed storage, Enigma [15] proposed a data computing scheme based on multi party computation (MPC), which theoretically solves the problems of data access control and encrypted computation. However, MPC has a great computational burden and can hardly support the general purpose computation of blockchain smart contract, its communication overhead is also very high, does not have the practical feasibility. Therefore, we propose a parallel distributed architecture based on these works. We adopt distributed storage to reduce the storage burden on the blockchain. At the same time, the built-in encryption scheme protects the privacy of data. The host of data cannot access to the original data. Besides, we design a complete data authorization and sharing scheme that ensure the confidentiality of data throughout the life cycle.

This paper is organized as following: Section II outlines the parallel distributed architecture and describe the design of PDash. Section III analyzes the security and performance of related techniques in PDash. Section IV illustrates application cases based on PDash. Section V concludes.

II. DESIGN OF PDASH

A. Architecture

PDash is a new architecture for blockchain systems which aims to solve the scalability problem at the data level. PDash separates the data and transaction on blockchain to release storage burden of nodes. Data produced by IoT systems is stored in the distributed storage nodes in the p2p network. Blockchain only holds the reference to data that serve as the identifier to verify the correctness and integrity of data. Blockchain is also responsible for access control of data stored in the cloud.

The decentralized system based on blockchain technology is different from traditional distributed system. In decentralized system, computing and storage tasks are redundant, which means every node in the decentralized network must store a complete copy of all data and perform the same operations of data. The redundant storage and computation on one hand make the blockchain system operates stably not relying on a trusted third party and ensure the integrity of data, the tamper-resistance of record and the consistency of system. On the other hand, too much redundant data also aggravates the burden of the system, which makes the new nodes' cost of joining the network more and more heavy. This approach is non-scalable and unsustainable in the long run. The Bitcoin blockchain stores only transaction data and has exceeded 160GB in size after nearly 10 years of operation. Ethereum realized the Turing complete virtual machine, supporting more kinds of smart contracts. In addition to the transaction data, the Ethereum blockchain also carried all the application data. In just two years, the blockchain size has surpassed bitcoin. New nodes have to consume a lot of time to synchronize blocks. Besides, the difficulty of new nodes to join continues to increase as time goes on.

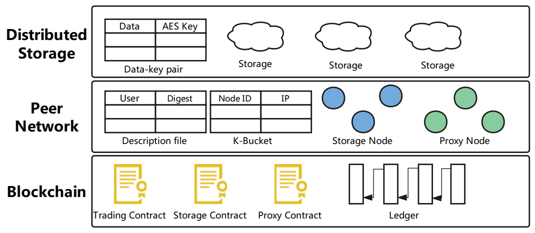

Fig. 1: PDash: Parallel Distributed Architecture for IoT Data Storage and Sharing

In this context, PDash proposes a three layer-architecture as shown in Fig. 1. The first layer is blockchain containing data identifiers and verification information of permissions. Blockchain is the control layer of the whole system and takes charge of validation of transactions and access control of data. It also provides environment for smart contracts to support applications like data trading. The second layer is peer network consists of millions of nodes who are willing to participate in this system. They can join or leave the system at any time and choose their roles as they wish. There are two special types of nodes, storage node and proxy node. Storage nodes provide their storage space for encrypted files produced by other nodes in exchange of token incentive. Proxy nodes provide their computation power to re-encrypt file from owner to consumer. Data owners delegate the decryption rights to consumers through re-encryption without revealing any plaintext to proxy nodes. Re-encryption simplifies the procedures for data trading. Data owner need not retrieve encrypted file from storage nodes, decrypt it, and encrypt the file once again under receiver's public key. They only delegate decrypt right to receivers through proxy nodes instead. All nodes maintain a description file to facilitate distributed search. Data owner adds a record consists of his account address and data digest into the description file when he upload a new data block. Nodes in the p2p network must update the description file periodically. The third layer is distributed storage layer. Data produced by IoT devices is stored in this layer after encrypted. At the same time, hash value of data is generated

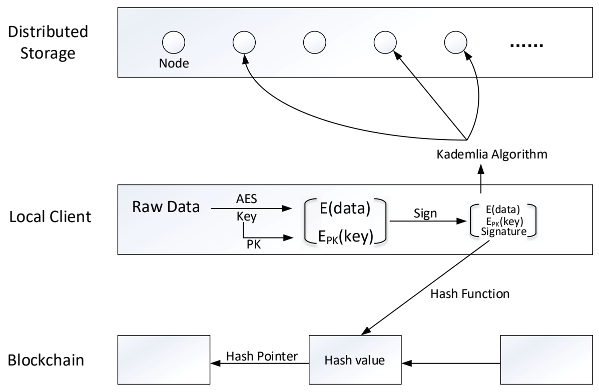

Fig. 2: DHT-based Distributed Encrypted Storage

and uploaded to blockchain as the unique identifier to data.

B. DHT-based Distributed Encrypted Storage

PDash separates the data layer from control layer. In data layer, we employ distributed hash table-based encrypted storage and achieve efficient sharing of encrypted data through re-encryption.

Fig. 2 shows the DHT-based distributed storage and encryption. All raw data is encrypted locally and stored in several storage nodes in the distributed network through Kademlia algorithm. The host of data have no access to original data as a result of encryption. Considering the balance of security and efficiency, we choose Advanced encryption standard (AES) as our symmetric encryption scheme. Each AES key only encrypts one file, then it is encrypted under owner's public key, packaged with the encrypted file, and stored in the distributed network together with the file. Besides, data owner also adds a digital signature of the digest of data block, which makes the ownership of data can be confirmed. There is no need for owner to store all AES keys whose size grows with time. Instead, owner only keeps the location information of key-file pairs. At the same time, blockchain also controls the access to data. Blockchain keeps records of access permission to every piece of data. Anyone who wants to retrieve the data must provide proof to host that meets the permission record.

The algorithm of data storage is shown in Algorithm 1. There are mainly four parts in the data storage algorithm. The first part (line 1 to 3) is the key generation including AES key and secret/public key pair (SK, PK). Initially data owner generates a random number and then produce the secret key. Public key is calculated from secret key. A one-time AES key is generated for each data block from master key. The master key and Secret key must keep offline and should never be accessed by others. The second part (line 4 to 6) is the encryption including data encryption and key encryption. Symmetrical encryption scheme is used in the data encryption and re-encryption scheme is used in the key encryption. AES key is encrypted

Algorithm 1 Data Storage

```
1: $sk_a \leftarrow random()$
2: $pk_a \leftarrow keyGeneration(sk_a)$
3: $key \leftarrow AESKeyGeneration(masterKey)$
4: $En(data) \leftarrow encryption(data, key)$
5: $En_{pk_a}(key) \leftarrow encryption(key, pk_a)$
6: $dataBlock \leftarrow (En(data), En_{pk_a}(key), sig)$
7: $target \leftarrow Hash(dataBlock)$
8: for $i = 1 \rightarrow k$ do
9:     $currentNode \leftarrow findNode(target)$
10:    if $Ping(currentNode)$ then
11:        $Store(currentNode, dataBlock)$
12:    end if
13:    $target \leftarrow target + 1$
14: end for
15: $identifier \leftarrow Hash(data)$
16: $Ledger(identifier)$
17: $DescriptionFile(des, addr)$
```

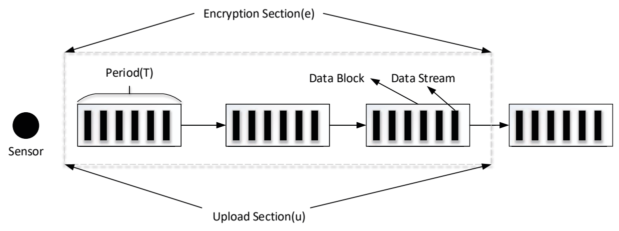

Fig. 3: Framework of Data Structure

under owner's public key, but it is different from traditional public key encryption. The re-encryption scheme based on bilinear map allows cipher text to be transferred with the permission of owner. The encrypted data and corresponding AES key are packaged with a digital signature. The third part is the storage (line 7 to 14). The data block is stored in the distributed network. The storage node are decided by the Kademlia algorithm, which means the k nodes whose node ID are closest to the hash value of the data block need to store a copy of data in the form of key-value pair. The last part (line 15 to 17) of the algorithm is calculating the hash value of data and uploading it to blockchain, and adding a description into the local description file. The hash value of data serves as the unique identifier of data for the verification of correctness and integrity of data. Note that storage nodes can deploy smart contracts for data storage, and the transaction of data storage can be completed through the smart contracts with incentive of token.

Both encryption and decryption of data consume computing resources. A huge number of data are generated by IoT systems all the time. Encrypting each piece of data individually is undoubtedly a tremendous waste of computing resources. Therefore, it is necessary to formulate specific data structure and encryption mechanism for different IoT systems to balance security and efficiency.

PDash arranges data in time sequence with chain structure, and set the period T to pack a period of data into blocks. On the basis of blocks, PDash chooses encryption section E and upload section U. Every E blocks perform an encryption and every U blocks calculate a hash certification uploading to the blockchain. A single record in the blockchain validates data in the whole upload section because of the chain structure. PDash sets different values of these parameters for different data types and application requirements. Fig. 3 shows an example of framework of data structures.

C. Proxy Re-encryption and Data Sharing

The separation of data layer and control layer solves the scalability problem of blockchain system at the data level, but also brings in another technical challenge of safe sharing of encrypted data. Data is encrypted and stored in the distributed network right after produced by IoT devices. All data is owned and controlled by users. Owner can delegate same data to multiple parties or different data to same party. So traditional public key encryption scheme no longer applies to PDash. We address this problem by proxy re-encryption technology. It allows proxy nodes to re-encrypt cipher-text for owner to cipher-text for receiver without accessing plaintext or secret key of owner.

The algorithm of data sharing is shown in Algorithm

Algorithm 2 Data Sharing

```
1: $target \leftarrow$
2: $consumer.query(DescriptionFile, keyWord)$
3: $consumer.sendTo(pk_b, target, provider)$
4: $rk_{a\to b} \leftarrow provider.reEncryptionKey(sk_a, pk_b)$
5: $permission \leftarrow (location, rk_{a\to b})$
6: $provider.sendTo(permission, proxy)$
7: $provider.Ledger(hash(permission))$
8: if $proxy.findValue(location)$ & $storage.checkPermission(Ledger, permission)$ then
9:     $storage.sendTo(dataBlock, proxy)$
10: end if
11: $En_{pk_b}(key) \leftarrow$
12: $proxy.reEncryption(En_{pk_a}(key), rk_{a\to b})$
13: $dataBlock \leftarrow (En(data), En_{pk_b}(key))$
14: $proxy.sendTo(dataBlock, consumer)$
15: $key \leftarrow consumer.decryption(En_{pk_b}(key), sk_b)$
16: $data \leftarrow consumer.decryption(En(data), key)$
17: $consumer.verification(Ledger, Hash(data))$
```

2. First, consumers search the local description file to find needed data, then they send requests to providers (line 1 to 3). The requests contain consumers' public keys and target data. In order to achieve more comprehensive search results, consumers must ensure that the description file is up-to-data, or they need to request latest data from neighboring nodes to update their own description file. Second, providers calculate the re-encryption keys according to their own secret keys and received consumers' public keys, then they send permissions containing location information of data and the re-encryption keys to proxy nodes as well as issue a transaction containing the hash value of the permission to the blockchain (line 4 to 7). Proxy nodes must provide the permission when they retrieve the data from hosts. Hosts allow proxy nodes to access the data only if the permissions they provide meet the records on the blockchain (line 8 to 10). Third, proxy nodes re-encrypt the AES key under providers' public keys to the ones under consumers' public keys (line 11 to 14). Note that proxy nodes can only re-encrypt data but can not decrypt it, and they can deploy smart contracts for re-encryption. The transaction of proxy re-encryption can be completed through the smart contracts with incentive of token. Last, consumers decrypt AES key by their own secret keys and then decrypt data by AES keys. Consumers verify the correctness and integrity of data through blockchain (line 15 to 17).

III. PERFORMANCE ANALYSIS

Currently we are developing the prototype of PDash mainly with python. The main modules include wallet, smart contract and storage. Wallet is the user interface that manage users' digital assets and keys. We are developing the wallet with PyQt5 and QML. Smart contracts are written in solidity on the Ethereum blockchain platform. We will develop a blockchain abstraction layer that interacts with Ethereum node through JSON-RPC. We will also encapsulate Ethereum's web3 interface to allow our wallet to call the APIs. As for storage part, both IPFS and Amazon S3 are optional. Data owners choose where they store their data. Virtual private server (VPS) is the best choice for proxy nodes, because VPS ensures proxy nodes are always online and data can be accessed at any time. This section is a preliminary discussion of several technologies. Our first release is under development. We will conduct a more detailed performance evaluation after it is developed.

A. Distributed Storage

1) Efficiency: PDash proposes parallel distributed architecture as the basis of the scalable blockchain platform for IoT data. We develop a distributed hash table based on kademlia algorithm. In the distributed network, each node maintains a table called k-bucket that contains IP addresses and node IDs of part of other nodes. Nodes in routing table are stratified by XOR distance, and there are k nodes on each layer. A node in the distributed network needs to maintain $160 \times k$ node information at most because the node ID is 160 bits. In the process of data retrieval, nodes search for target node according to the XOR distance. Relay nodes forward the request until it reaches the target node. The distance is shortened by half at least at each forwarding. Therefore, the retrieval of any data needs n steps at most in the network with 2^n nodes.

2) Security and Privacy: The storage node can not access the original data because all the data is encrypted on the client side. Nodes without permission cannot retrieve the data even they have known location information of the data by some means as a result of the access control of blockchain. Another concern is that some storage nodes may ignore the verification on the blockchain and hand over data to nodes without permission. However the effect of this kind of dishonest behavior is limited because all data stored in the storage nodes is encrypted. Storage nodes have token incentive if they provide storage service honestly, they have no motivation to fraud.

B. Encryption

All data on the PDash are encrypted locally and keep an encrypted state in the whole process of storage and sharing. We employ AES as symmetric encryption scheme considering the limitation of computing resources of IoT devices. AES is based on a design principle known as a substitution-permutation network, a combination of both substitution and permutation, and is fast in both software and hardware with a low consumption of computing resources. We choose 128-bit keys to reach a balance of security and performance. The number of possible combinations is 3.4×10^{38}that is secure enough. At the same time, each AES key is used only once for a higher level of security and more specific data sharing.

PDash realizes multiple delegations of encrypted data through the re-encryption technology. The most advanced re-encryption scheme has already achieved following important features [16].

1) Unidirectional: The delegation is unidirectional. Data owner cannot re-encrypt data under receiver's public key when he delegates the decryption right to the receiver.

2) Collusion-safe: A main concern of re-encryption is that by colluding proxy node and receiver of data can recover data owner's secret key. Our re-encryption scheme introduces a weak secret key to avoid this risk. For example, suppose owner's public key is $e(g, g)^a$ and his secret key is a. In our system, even proxy node and receiver collude, they can only recover the value g^a, but not the a itself.

3) Non-transitive: The proxy node cannot re-delegate the decryption right without another permission. Even the proxy node possesses $rk_{A \to B}$ and $rk_{B \to C}$, it cannot generate $rk_{A \to C}$.

C. Blockchain

Blockchain only stores digests of data, but not data itself in the new architecture. The amount of data on the blockchain is greatly reduced with the PDash. PDash allows new nodes to join the system easily without having to synchronize all data for all applications. This feature is important for IoT system, because the amount of data generated by IoT system is huge. PDash makes it possible to introduce blockchain technology into IoT field. Besides, TPS (transactions per second) is the main bottleneck problem of current blockchain system. Increasing block size is a main-stream solution, but simply increasing block size improves maintenance cost of nodes, resulting in fewer nodes and lower system security. In PDash, the block size is constant, while the number of transactions that can be packaged into a single block can be greatly increased, which enhances the transaction rate greatly.

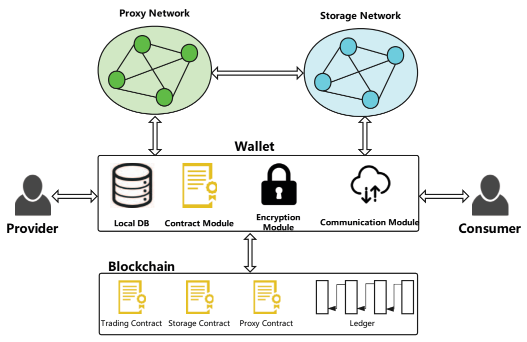

Fig. 4: Distributed Data Sharing Platform

IV. USE CASE

A. Distributed Data Sharing Platform

Data is produced, owned and controlled by users in our system. Token is a core component in this ecosystem. A multi-party distributed data sharing platform can be built based on token. Provider, consumer, storage node, proxy node and smart contract, all participants perform their duties as shown in Fig. 4.

There are four entities who participant in the distributed trading market for digital assets. The interactions between providers and consumers are completed with the help of wallet, a client that manages users' digital assets and handles local communication with blockchain, storage network and proxy network. There is no centralized trusted third party standing in the system. Storage Nodes deploy storage contracts on the blockchain, and provide storage space for data owners to earn token. Data owners are the providers in the market. Providers store their IoT data at the storage network and upload the references to data to blockchain through the wallet. Providers need to deploy trading contracts and add descriptions of data into local description file when they want to sell their data. Consumers update their local description file firstly, then search the file to find needed data. After that, consumers request data from owner. providers will not send the data to consumers directly because they do not save all data locally. Trading is completed by providers delegating the decryption right to consumers through proxy nodes. Proxy nodes are similar to storage nodes. Proxy nodes provide their computation power to re-encrypt data for provider to the one for consumer with permission from provider. Note that, storage nodes will check the authenticity of permissions by blockchain when proxy nodes try to

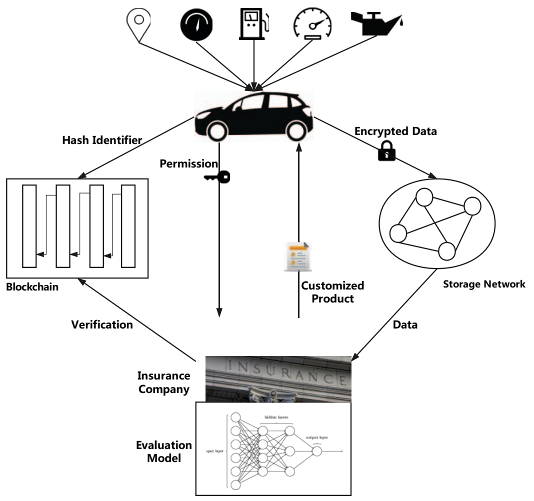

Fig. 5: Personalized Auto Insurance Scheme

retrieve encrypted data. Consumer can also ensure the correctness and integrity of data by blockchain. Besides, when a dispute between the consumer and the provider occurs, the proxy node will participant in the ruling as a witness. In order to ensure the transaction, an intermediary must be introduced. Another advantage of introducing proxy nodes is the reduction of user computing load. For old consumers, the re-encryption keys are already reside in proxy nodes. Besides, there are more specific functions that can be added at the proxy layer, such as limits of the access time and times of download.

B. *Personalized Auto Insurance*

Auto insurance is essential for drivers, but the degree of customization of existing insurance schemes is very low due to the choices are similar for novices and experienced drivers. it is not cost efficient for drivers. At present, it is not feasible for insurance companies to customize different insurance schemes for different drivers because of the lack of sufficient data. It is difficult for insurance company to assess drivers' driving habits accurately. Drivers can collect and store a large number of their vehicle data at PDash. Companies are able to perform better assessments of drivers' driving styles according to modeling and analysis of these large amount of data. Besides, Insurance for single trip can be generated and provided to users automatically in this model.

Fig. 5 illustrates the PDash-based auto insurance service model. Vehicle sensors collect a lot of driving data and aggregate them to data gateway. Data gateway organizes and encrypts sensor data and upload them to the distributed storage network. At the same time, the hash identifier of these data is uploaded to the blockchain network. The data will be authorized to the insurance company when user wants to customize auto insurance product. The insurance company downloads the data from the distributed storage network and decrypts them, and verifies the authenticity and integrity of the user data through blockchain. After that, the driver?s rating is assessed according to the evaluation model, and the personalized insurance product is provided to user.

V. Conclusion

This paper presents PDash with parallel distributed architecture for infrastructure of IoT data based on blockchain, analyzes performances of several crucial techniques and designs decentralized applications to verify its feasibility. The aim of PDash is to improve scalability of current blockchain systems. PDash separates data layer from blockchain layer, blockchain only store references to data but not the original data to be responsible for access control and data authorization. Data storage layer and blockchain control layer interacts securely through encryption techniques such as symmetric encryption, and re-encryption. The parallel distributed architecture has the full potential to be the infrastructure of the Internet of Things.

Acknowledgement

This work was supported in part by the National Natural Science Foundation under Grants 61673275 and 61473184.

References

[1] G. Zyskind, O. Nanthan and A. S. Pentland, "Decentralizing privacy: Using blockchain to protect personal data," in *IEEE Security and Privacy Workshops (SPW)*, 2015, California, USA, pp. 180-184.
[2] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.
[3] S. Bano, A. Sonnino and M. Al-Bassam, et al., "SoK: Consensus in the Age of Blockchains," arXiv preprint arXiv:1711.03936, 2017.
[4] S, King, S, Nadal. "Ppcoin: Peer-to-peer crypto-currency with proof-of-stake," self-published paper, 2012.
[5] I. Bentov, C. Lee and A. Mizrahi, et al. "Proof of Activity: Extending Bitcoin's Proof of Work via Proof of Stake[Extended Abstract]y," *ACM SIGMETRICS Performance Evaluation Review*, vol. 42, no. 3, pp. 34-37, 2014.
[6] Y. Gilad, R. Hemo and S. Micali, et al. "Algorand: Scaling byzantine agreements for cryptocurrencies," in *ACM Proceedings of the 26th Symposium on Operating Systems Principles*, 2017, Shanghai, China, pp. 51-68.
[7] S. Micali, "Fast and furious Byzantine agreement," in emphProceedings of the Innovations in Theoretical Computer Science (ITCS), 2017, Cambridge, Massachusetts.
[8] R. Pass and E. Shi, "Hybrid consensus: Efficient consensus in the permissionless model," in emph31st International Symposium on Distributed Computing, 2017, no. 39, pp. 1-16.
[9] J. Poon and T. Dryja, "The bitcoin lightning network: Scalable off-chain instant payments," 2016.
[10] L. Luu, V. Narayanan and C. Zheng, et al., "A secure sharding protocol for open blockchains," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016, Vienna, Austria, pp. 17-30.
[11] S. Popov, "The tangle," 2016.
[12] J. Benet, "Ipfs-content addressed, versioned, p2p file system," arXiv preprint, arXiv:1407.3561, 2014.
[13] S, Wilkinson, T. Boshevski and J. Brandoff, et al., "Storj: a peer-to-peer cloud storage network", 2014.
[14] Whitepaper, "Filecoin: A Decentralized Storage Network," https://filecoin.io/filecoin.pdf, 2017.
[15] G. Zyskind, O. Nathan and A. Pentland, "Enigma: Decentralized computation platform with guaranteed privacy," arXiv preprint, arXiv:1506.03471, 2015.
[16] G. Ateniese, K. Fu and M. Green M, et al., "Improved proxy re-encryption schemes with applications to secure distributed storage," emphACM Transactions on Information and System Security (TISSEC), vol. 9, no. 1, pp. 1-30, 2006.