NormaChain: A Blockchain-Based Normalized Autonomous Transaction Settlement System for IoT-Based E-Commerce

Chunchi Liu, Yinhao Xiao, Vishesh Javangula, Qin Hu, Shengling Wang, *Member, IEEE*, and Xiuzhen Cheng, *Fellow, IEEE*

***Abstract*—Internet of Things (IoT)-based E-commerce is a new business model that relies on autonomous transaction management on IoT-devices. The management system toward IoT-based E-commerce demands autonomy, lightweight, and legitimacy. As blockchain is an innovative technology that is competent in governing the decentralized network, we adopt it to design the autonomous transaction management system on IoT E-commerce. However, current blockchain solutions, most namely cryptocurrencies, have fatal drawbacks of nonsupervisability and huge computational overhead, and hence cannot be directly applied on IoT-based E-commerce. In this paper, we propose NormaChain, a blockchain-based normalized autonomous transaction settlement system for IoT-based E-commerce. By designing a special three-layer sharding blockchain network, we can significantly increase transaction efficiency and system scalability. Additionally, by designing an innovative decentralized public key searchable encryption scheme (decentralized public key encryption with keyword search (PEKS) scheme), we can uncover illegal and criminal transactions and achieve crime traceability. Our new decentralized PEKS scheme cryptographically eliminates the dependence of a trusted central authority in the original PEKS scheme and instead expands it to a fully decentralized governance, which distributes the supervision power equally among all parties. More importantly, by proving NormaChain is secure against chosen ciphertext attacks and against the stealing of the secret key, we show that NormaChain prevents a legitimate user's privacy from being violated by banks, supervisors or malicious adversaries. Finally, we deliver the NormaChain system with design details and full implementations. Experiments show that the average transaction-per-second on IoT devices is around 113, and the supervision accuracy is 100% when proper target illegal keywords are provided.**

***Index Terms*—Blockchain, Internet of Things (IoT)-based E-commerce, public key encryption with keyword search (PEKS), searchable encryption (SE), sharding, smart contract.**

Manuscript received June 1, 2018; revised September 25, 2018; accepted October 12, 2018. Date of publication October 23, 2018; date of current version June 19, 2019. This work was supported in part by the U.S. NSF under Grant IIS-1741279 and Grant CNS-1704397 and in part by the National Natural Science Foundation of China (No. 61472044, No. 61772080). *(Corresponding author: Qin Hu.)*

C. Liu, Y. Xiao, Q. Hu, and X. Cheng are with the Department of Computer Science, School of Engineering and Applied Sciences, George Washington University, Washington, DC 20052 USA (e-mail: liuchunchi@gwu.edu; xyh3984@gwu.edu; qinhu@gwu.edu; cheng@gwu.edu).

V. Javangula is with the Department of Biological Sciences, School of Engineering and Applied Sciences, George Washington University, Washington, DC 20052 USA (e-mail: visheshj123@gwmail.gwu.edu).

S. Wang is with the College of Information Science and Technology, Beijing Normal University, Beijing 100875, China (e-mail: wangshengling@bnu.edu.cn).

Digital Object Identifier 10.1109/JIOT.2018.2877634

I. Introduction

INTERNET of Things (IoT) is a collective set of technologies that connects and organizes a network of lightweight devices. Recently, the concept of IoT-based E-commerce is emerging as a new trading model, which realizes person-to-machine (P2M) or even machine-to-machine (M2M) transactions, rather than person-to-person (P2P) transactions as in the conventional E-commerce. For example, Amazon Dash is a one-click button that automatically purchases the assigned product. This is a typical example of extending the E-commerce from P2P to P2M transaction model. Additionally, CEO of JD.com, the Chinese E-commerce giant, has just made a full autonomous commitment, expecting robotics and M2M algorithms to eventually take over its supply chain. In addition, in 2017 JD successfully built the world's largest fully autonomous logistics center in Shanghai. It is therefore reasonable to imagine the future of E-commerce, where all levels of settlements are completed in a purely autonomous and M2M fashion.

However, current IoT-based E-commerce systems are often constructed with a crowd of fragmented and lightweight IoT devices. To govern this scattered structure, an autonomous, accountable, and lightweight M2M framework must be deployed. Blockchain's ability on governing decentralized networks makes it especially suitable for designing a self-management system on IoT devices. Guaranteed by rigorous cryptography, current blockchain solutions can establish trust and run in a self-governed way without the need of a central authority. Additionally, its hash-connected chain data structure achieves almost-perfect data integrity. Transaction data can thus be stored and shared with great confidence. Further enhancement of IoT device autonomy can be done using smart contracts, which serve as digital contracts reinforced by codes.

Cryptocurrency, the signature blockchain application that supports numerous E-commerce systems, however, cannot be directly applied to the IoT-based E-commerce. For example, the iconic Bitcoin established their reputation by offering perfect transaction anonymity and security. However, these impressive features are gained in sacrifice of legitimacy and efficiency. As for legitimacy, it is often abused by fueling the digital black markets such as the "Silk Road," who traded $1.2 billion worth of transactions in 2013 [1]. This symbol of disorder is a major reason that blockchain is feared and rejected by governments and industries. Furthermore, its transaction

security is maintained at the cost of huge hash power, which causes high computational overhead. These critical drawbacks of blockchain are fatal to the future IoT-based E-commerce as they continue to deter government and industry confidence.

In order to perform necessary legal supervision on transactions, regulators must obtain user's transaction information to a certain degree. For example, regulators may need to decrypt a suspicious transaction history to uncover any illegal or fraudulent transactions. Some systems address this issue by storing transaction records in plaintext, however, this infringes on the user's privacy and can nullify the privacy promised by blockchain. However, if we do not allow such inspection by keeping the records encrypted and identity anonymous, we face the same anarchy problem. Currently, blockchain communities seem to be trapped in a dilemma where they have to choose between privacy and legitimacy. To the best of our knowledge, NormaChain is the first work to settle the above dilemma and preserve both essential features. To be more specific, we propose a legal supervision scheme on blockchain using searchable encryption (SE).

SE is a technique that permits search for specific target keywords on a piece of encrypted data without revealing the plaintext. As a result, we can search for specific criminal evidence while keeping the legitimate customers' information untouched. According to the needs of our scenario, we adopt the public key encryption with keyword search (PEKS) scheme [2]. Most importantly, one of our major contribution in this paper is that we propose a decentralized PEKS (DPEKS) scheme. In this system, we eliminate the need of a central authority to perform supervision, which was needed in the original PEKS scheme. Instead, we distribute the power of supervision to n different parties to avoid single point corruption. Only when the target keyword list is approved by all n parties simultaneously, can this target keyword be allowed to search. This provides a high security to tolerate any kind of collusion under $n-1$ parties' corruption. In the security analysis part in this paper, we show that NormaChain preserves legitimate user's private information against the supervisor by proving NormaChain is CCA2-secure. This means that supervisors cannot attack the system and infer the encryption key and thus decrypt an arbitrary user's encrypted information. Furthermore, we also prove that in our decentralized computing process, our secret key is kept safe from being cryptanalyzed.

Progressively, our autonomous and lightweight transaction management platform is designed for IoT E-commerce to address the efficiency challenge. In the goal of achieving high transaction speed and scalability, we replace the conventional single layer blockchain to a three-layer sharding blockchain network, with each layer assigned to different responsibilities. We also adopt practical byzantine fault tolerance (PBFT) consensus algorithm in replacement of proof-of-work (PoW) to minimize the overall mining liability of our nodes.

We lastly present our NormaChain system with full detailed design and implementation over C++. Our experiment result shows that when hosting on a consumer-grade laptop, or even on lightweight IoT devices (Raspberry Pi), the transaction latency remains on the millisecond level and the supervision can also be executed in real time. Although the transaction speed is slightly increased due to the weak computational ability of these devices, our lightweight system still takes no more than 8 ms to complete a transaction. The average transaction-per-second (TPS) of NormaChain on IoT-devices is 113.69. Supervision accuracy reaches 100%.

The rest of this paper is organized as follows. In Section II, we first introduce the related work. Next, we explain the necessary preliminaries in Section III. In Section IV, we present the three-layer NormaChain architecture and the adversarial model. In Section V, we deliver our DPEKS scheme. The detailed system implementation is presented in Section VI. In Sections VII and VIII, we elaborate the security analysis and evaluation results. We conclude this paper in Section IX.

II. RELATED WORK

The most related researches to this paper can be summarized into two perspectives, IoT-based E-commerce and blockchain applications in E-commerce.

IoT-Based E-Commerce: The core concept of IoT-based E-commerce is to establish M2M communication model, in replacement of the traditional P2P model to the maximum extent. Huang *et al.* [3] defined the M2M communication model to be an automated communication process among machines with minimum human interventions. As IoT-based E-commerce prevails in recent years, numerous applications begin to emerge. Card-based online digital payment system is the first generation of IoT-based E-commerce that has supported and dominated the market for a long time. JW model [4] proposed by Asokan [5] and 3e model [6] were widely employed. Although this form of digital payment permits some degree of convenience, it is merely an extension of the traditional credit card, and likewise, suffering significantly from fraudulent transactions and high latency. According to the survey conducted by Kiernan, the world suffers from a $21.8 billion loss on credit card fraudulent transactions, in 2015 alone [7].

Furthermore, credit cards are currently limited to a P2P or P2M trading model, and cannot support M2M autonomous transaction management. Later with the popularity of mobile devices, the mobile payment system has become one of the most critical components of IoT-based E-commerce. Platforms like Paypal or Venmo have been gaining more and more attention from the public. González [8] took PayPal as an example to analyze the digital payment system in detail. Gao *et al.* [9] even took a step forward by proposing a P2P payment system, named P2P-Paid, enabling two users to transfer money through Bluetooth communications. However, these systems function as an extended "buffer" of banks, and still cannot achieve autonomous M2M settlement management. Blockchain-supported IoT-based E-commerce system, which owns the features of data integrity, nonrepudiability, and autonomy is a satisfying solution that serves as a secure, traceable and autonomous transaction management system.

Blockchain Applications in E-Commerce: Bitcoin is the first pioneer and iconic blockchain application in E-commerce that achieved practical data integrity and perfect transaction

anonymity [10]. Following Bitcoin, there evolved numerous optimizations and innovative designs of digital payment systems. Roos *et al.* [11] proposed a path-based transaction (PBT) method to further improve efficiency, while preserving high success ratio of processing a transaction. Pass and Shi [12] achieved an instant transaction-confirming state machine replication, by combining a fast, asynchronous path with a (slow) synchronous "fall-back" path. Kokoris-Kogias *et al.* [13] ensured security and correctness by using a bias-resistant public-randomness protocol for choosing large and statistically representative shards that process transactions, and further improved the scalability using cross-shard commit protocol. These applications and optimizations helped popularize the use of public/private ledgers outside of the cryptocurrency market and shifted the public focus on using blockchain technology for constraining transaction forgeability and providing secure M2M communications.

Attempts in enhancing blockchain traceability and autonomy in the supply chain have also been made. Tian elaborated a case study on a successful and innovative management of agricultural products using blockchain in China [14]. However, it still lacks a solution to autonomously generate a transaction, settle the finance, and connect seamlessly to the supply chain. Xu *et al.* [15] proposed an initial solution to this problem by using a Bitcoin wallet and embedded crypto chip hardware. While these additions automated gas payments, the absolute anonymity of Bitcoin continues to enable online black markets and criminal trades through sites such as the Silk Road [1].

Other works embraced supervision by allowing open access to their blockchain systems for authorities figures such as governments. Chen *et al.* [16] designed a blockchain-based payment collection supervision system (BPCSS), aiming to include supervisions from governments and regulatory agencies, while maintaining the benefits of a decentralized network for E-commerce. Using cloud databases to store business and consumer information, this system used Bitcoin and blockchain coupled with servers to verify that a transaction was properly executed. Governments and regulatory agencies were granted administrative access to these databases, which housed all the buyers and sellers information and transaction histories. Such direct access to all consumer information compromises user privacy, and therefore, is unlikely to be accepted by general consumers. Having realized this deficiency, we deliver our design of NormaChain: a lightweight, transparent, autonomous E-commerce settlement and management system enhanced by privacy-preserving supervision for combating illegal transactions. These features are discussed with details in the rest of the contexts.

III. Preliminaries

Prior to introducing NormaChain, we first present background knowledge on the methods we intend to use. For this project, blockchain serves as the backbone of the autonomous transaction management system, and SE is used as the core method of carrying out privacy-preserving supervision.

A. Blockchain System

Blockchain is a short and iconic term referring to a consensus-supported decentralized network. It relies on pure cryptographic designs to achieve trust, security, and privacy without a central authority. The blockchain network is constituted with basic structures and functions as follows.

1) *Blockchain Data Structure:* The hash-connected data structure that permits only adding new information (block) at the back, and forbids modification on the past information. The integrity and protection of the past information are ensured by a consecutive hash that tightly connects every adjacent block in a sequence of time. Nodes in the blockchain network can easily validate all blocks' integrity through this special structure. In this paper, blocks will store transactions and trader information.
2) *Consensus Algorithm:* A kind of algorithm that unifies all nodes in a network to achieve a universal agreement. This replaces the central authority as a source of trust. The agreement, or consensus, is a critical decision and should be stored in the block as history. Specifically, in blockchain, this consensus is usually about the transaction details. Some most influential consensus algorithms include PoW, proof-of-stake (POS), delegate POS, paxos, ripple protocol of consensus algorithm, PBFT, and so on [10], [17], [18].
3) *Distributed Ledger:* A local ledger of a node in the blockchain network that stores important information. In most blockchain system designs, the ledger is an exact copy of the blockchain and is stored locally by each node. This ensures that all stored data are protected by the crowd, rather than a single central authority. As a result, any attempt to compromise a single node can easily be recognized and resisted. In some designs, distributed ledger stores rules or information that are also critical to the system. But all of them should be public to all other nodes for transparency.

In the last few years, the blockchain has evolved significantly since the basic 1.0 version in the Bitcoin age. Rather than only being utilized in the cryptocurrency field, it is now widely perceived as a general concept of designing a transparent and reliably shared network. Consequently, there have been numerous improvements for expanding its scalability and general performance. For example, sharding is a widely implemented idea that divides the blockchain network into different layers each with a respective job. It has been experimentally shown to be useful in improving transaction speed and stability as well as reducing data redundancy [19]. In this paper, we adopt the sharding blockchain technology to design our NormaChain network.

B. Searchable Encryption

SE is a theory of carrying out reliable and privacy-preserving searching on encrypted data. Although it permits the use of an agent to search through sensitive data, excess leakage of information is restricted due to SE's architecture. It is worth mentioning that since blockchain records are

protected by encryption, SE's functionality of searching over encrypted data is equivalent to that of online search engines used on the traditional Internet infrastructure. SE was first introduced by Song *et al.* [20] who is credited with developing Symmetric SE. In this primitive scheme there are two cryptographic roles: 1) Alice and 2) Gateway. Alice is the data writer and owner who holds a private key. The data and trapdoor are both encrypted and constructed with this private key. Alice can thus grant keyword search permission to the Gateway for identifying a desired keyword, however, the Gateway remains unaware of the encrypted text's contents. This scheme only permits one data writer and owner, and is therefore called single writer–single reader SE scheme. This is not suitable for our system, as the blockchain is possibly written, shared, and maintained by all nodes, which lead us to a progressive scheme of the PEKS, which is published by Boneh *et al.* in 2004 [2].

In PEKS, there are three roles instead of two, i.e., Alice, Bob, and the Gateway. Alice is the data and private key holder, while Bob holds only the public key of Alice and thus can write and encrypt the data, and the Gateway is the agent of searching. In this scheme, both Bob and Alice can write the data with the public key, but only Alice can grant permission to search with the private key. This scheme is thus called multiple writer–single reader (M/S) SE scheme, which supports multiple different entities to write the data but the only one can search. In this case, it is suitable to employ PEKS for the data sharing the purpose of the blockchain system. The ability to search is achieved by the construction of trapdoor. A trapdoor is a tool that can determine whether the encrypted information contains a desired keyword while maintaining the texts encrypted nature. However, one can see that Alice has the monopoly power of deciding what to search. This is reasonable if the data is owned purely by Alice, but when a third-party uploads data, their right to search for an illegal keyword should be granted. We thus adopt the cryptographic core of PEKS in NormaChain and further design a DPEKS scheme, where our special needs can be fulfilled.

IV. ARCHITECTURE AND ADVERSARIAL MODEL

In this section, we introduce the architecture and adversarial model of NormaChain. We first introduce the three-layer blockchain network design of NormaChain, then followed by the essential roles in NormaChain and the specific adversarial model.

A. Three-Layer Sharding Blockchain Network Design

This network model complies with the hierarchical design concept, which divides the blockchain network into three layers, each being responsible for different tasks. This hierarchical design can significantly improve system performance of blockchain as it assigns tasks by taxonomy, and distributes them to the best corresponding workers [21]. This design consists of transaction, approval, and supervision layers, as demonstrated in Fig. 1. In the whole blockchain system, we have two chains, i.e., the transaction chain and supervision chain, where the former is shared but encrypted in the

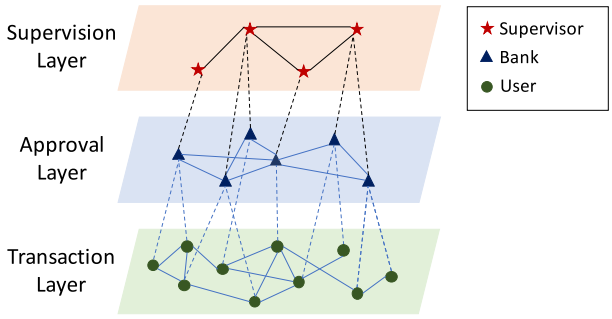


Fig. 1. Three-layer sharding model of NormaChain.

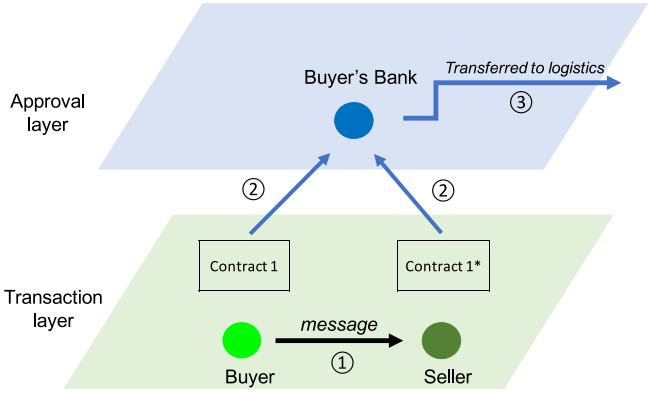


Fig. 2. E-commerce contract generation and approval diagram.

transaction and approval layer, and the later is encrypted and kept private in the supervision layer.

The lowest layer is a public transaction layer consisting of users that are either online buyers and/or E-commerce merchants. All user nodes, denoted by U_i, can freely connect and disconnect to the network, as they are not required to carry out any mining or verification duties. Whenever any user U_i initiates a transaction TX_i, two identical contracts between the buyer and seller are automatically generated and sent to the middle layer's connected banks for approval (Fig. 2). Therefore, the transaction layer does not have any mining liabilities, nor does it has to store the full transaction chain as those duties are transferred to the approval layer. Users are only responsible for initiating transactions and sending it to the buyer's bank for further processing.

The middle layer is a consortium approval layer. Nodes in this layer are financial institutes, denoted as B_i for banks, which can only verify financial transactions for their users' transactions. These interconnected nodes authenticate B_i's identity and authorize him to verify the generated transaction contract TX_i. Once a consensus is reached, B_i can thus verify the transaction contract, encrypt it with the public key β, and push the ciphertext C_{TX_i} onto the transaction chain. In this way, the transaction of each user is only known by his corresponding bank and hence user's privacy is not shared to other banks in the approval layer. Note that the public key and the private key pair will be generated distributedly. The public key is revealed to the public as we need it to encrypt messages,

but the private key is scattered into $|B_i|$ pieces and distributed to every bank B_i. We detail the specific steps in Section VI.

The upper layer is a private supervision layer. All nodes in the supervision layer are authorities that require an invitation to join. These supervisors, denoted as S_i, could be government agencies, law enforcers, NGOs, etc. Within a period of time, the supervisors can propose to scan a target illegal keyword list w. All the banks will collaborate to calculate the corresponding trapdoor T_w, if all of them consider this keyword list w is reasonable and nonprivacy-violating. All attempts to scan the transactions are recorded on the supervision chain to ensure the accountability of supervision power. Only if any illegal information or keywords are spotted can it be picked out for further inspection.

B. Adversarial Model

We mentioned earlier that there are three cryptographic roles in the PEKS scheme, Alice, Bob, and Gateway. In our scenario, all banks B as a whole virtually plays the role of Alice by decentralizedly holding a private key α through a cryptosystem. The public key β is publicly shared to all banks. Any bank B_i who uses Alice's public key β for encrypting approved transactions TX_i is Bob. Any supervisor S_i can be the Gateway with a criminal target keyword list w by requesting trapdoors from B. Thus, supervisors can search for criminal information without knowing the full contents of user transactions. In this paper, we emphasize the importance of security and privacy of all parties. As a result, we analyze and propose the adversarial models for all parties in NormaChain.

1) Adversarial Supervisor: The role of supervisor in our system is one major source of adversarial attacks, as government agencies sometimes tend to peek inside the full information of a customer. So the supervisor adversarial model in this paper is set as "honest but curious," which means the supervisor would perform his duties, but stays curious on full customer's plaintext information. Again, a major feature of this paper is to successfully permit a general privacy-preserving supervision on the blockchain systems to address the aforementioned problem. In Section VII security analysis, we provide the proof of chosen ciphertext security attack (CCA) against the supervisor.

2) Adversarial Bank: The adversary model of banks is similar to the adversarial supervisor. They can be honest but curious but this time, they can curious on recovering the complete secret key, such that one can decrypt the encrypted information, or compute any trapdoor he wants. For any bank, he can collect all public information, such as fractional trapdoor T_{w_i}, public key etc. He can also try to collude with other banks to share their private information to make a malicious impact on the whole. Yet in Section VII, we provide rigorous proof of impossibility to cryptanalyze, recover and obtain other bank's secret key share from all accessible information. Even by collusion, it is extremely hard to obtain enough information for malicious activities.

3) Adversarial User: Since the users in our system bear minimal responsibilities for maintaining the transaction chain (no mining liabilities), the user adversarial model does not have much power to perform malicious activities. The adversarial user model in this paper is set to tamper with the transaction contract and his balance [22]. This is also easily defensible with blockchain integrity property.

V. Decentralized PEKS Scheme

After initializing the three-layer blockchain network and adversarial model, we now provide our design of DPEKS on it. As discussed earlier, we adopt PEKS for NormaChain, as it can support M/S scheme on encrypted data. We must point out that the original PEKS scheme relies on a centralized governance of the private key, and thus have the ability to secretly construct unnecessary trapdoors, or peek into the encrypted data. Of course one can assume the integrity and the honesty of the central authority, however it is easy to see that this assumption is less than vulnerable. In our design, we further expand the PEKS scheme into a pure decentralized way—no one has the full access to the private key, while still preserve the ability to collaboratively computing the required output, i.e., trapdoor. We call this new scheme the DPEKS. This is achieved by combining the distributed key generation (DKG) scheme and a delicate expansion of the original PEKS scheme. We also give careful proof of correctness and security of our new scheme.

A. Distributed Key Generation

The first DKG scheme was introduced by Pedersen in 1991 [23]. It is a delicate design to simultaneously assign n different secret key shares α_i to n different people. The overall secret key α is virtually shared among all parties as a mathematical combination of all α_i. In this manner, we can promise that no single party can reconstruct or store the secret key on his own, and the secret key can only to be used by decentralized collaboration. Thus, we can effectively eliminate the need of a centralized trust. This is also critical for our DPEKS scheme.

One of the most commonly used DKG scheme is $\mathsf{Joint-Feldman}$ protocol [24]. It is basically letting n players to simultaneously run the Feldman's information-theoretic Verifiable Secret Sharing protocol, and finally assign n secret keys shares α_i to n different people. The complete protocol is page consuming, so we simplify it by omitting the details such that it is more accessible to all readers. Here, we have the simplified DKG algorithm defined as follows:

$$\begin{aligned} \mathsf{DKG}(n) &= \{\alpha_i\} \\ &\alpha_i \in \mathbb{Z}_q, i \in \{1, \ldots, n\} \end{aligned} \tag{1}$$

where g is the generator of finite cyclic group $\mathbb{Z}_q$ of q order, p is a large prime number and q is a large prime dividing $p-1$. The generator g, order q and p are necessary and must be identical to every participant, so they are presettled by the protocol, and distributed to all. The generated secret key shares α_i should satisfy the equation of $\alpha = \sum_{i=1}^{n} \alpha_i$. Each piece of secret key share α_i can never be revealed to others, or will be considered as collusion.

As the public key is g to the power of the secret key, so according to the definition of the cyclic group, the overall

public key h can be collected as

$$h = \prod_{i=1}^{n} h_i = \prod_{i=1}^{n} g^{\alpha_i} \tag{2}$$

and explicitly shared to all.

We clarify that we do not need the explicit revealing of private key α to perform regular functions. In fact, we prohibit any party to reveal his own secret key share to anyone (which will be seen as collusion) and hence the secret key α should never be reconstructed explicitly—unless all n parties are corrupted by the adversary at the same time.

The question here is: if we prohibit explicit reconstruction of the secret key α, then how can we use it? Note that the only place we need to use the secret key α as a whole in the original PEKS scheme is when we compute the trapdoor against illegal keywords. We now prove it in our DPEKS construction at the following section, that even with the fractions of the secret key $\{\alpha_i\}$, one can locally build its fractional trapdoors T_{w_i}, which can be further linearly collected as the final functional trapdoor T_w. The correctness and secrecy of trapdoor are proven in the later context.

Normally, when the supervisor proposes to search for an illegal keyword list w, all banks will evaluate the necessity of it. If all banks agree that this list is reasonable, they will collaborate to compute the trapdoor. Otherwise the keyword list might be harmless and unnecessary, then the bank can decide to accept or reject—each bank will have its own opinion on customer protection.

B. DPEKS Scheme

Here, we have all the prerequisites we need. We can thus propose the DPEKS and its formal definitions.

Definition 1: A DPEKS search scheme should consist of the following polynomial-time algorithms.

1) DisKeyGen*(n):* Input the total number of n parties, generates a set of secret key shares $\alpha_1, \alpha_2, \ldots, \alpha_n$, each hold secretly by the player P_i, and a corresponding public key $\beta = g^{\sum \alpha_i}$ is broadcasted to all.
2) DPEKS*(β, W):* Input the public key β and a transaction W, produces an SE of W.
3) TrapdoorLocal*(α_i, w):* Input the secret key share α_i and a target wordlist w, outputs a fraction trapdoor list T_{w_i}.
4) TrapdoorCollection*($\{T_{w_i}\}, T_{w_j}$) $\rightarrow$ T_w :* Receives all fraction trapdoor T_{w_i} from other players, use his own secret key share α_j to locally compute $T_w = \prod_{i \in n \setminus j} T_{w_i} \cdot T_{w_j}$ (where $\cdot$ is the group operation), and broadcasts for verification.
5) Test*(β, C_i, T_{w_i}):* Input the public key β, an SE S = DPEKS(β, W_1), and a trapdoor T_{w_2}. Outputs "YES" if $W_1 = W_2$ and "NO" otherwise.

Compared with the original PEKS scheme that was proposed by Boneh *et al.* [2], our scheme eliminates the need of a central holder of the secret key α. Rather, we generate secret key shares α_i in a distributed manner, decompose the computation of fractional trapdoors T_{w_i} in the local side and finally collect them into the complete T_w. This can be reflected as the difference from the original KeyGen(), to our new DisKeyGen(), and the Trapdoor() to become TrapdoorLocal() and TrapdoorCollection(). As a result, we must guarantee that the DPEKS scheme is correctly decomposable and collectable. Here, we define the correctness of DPEKS as follows.

Definition 2 (The Correctness of DPEKS):

1) The sum of all secret shares α_i is the unique secret key α, and all parties have the same public key of the value $h = g^{\alpha}$.
2) The Trapdoors are correctly computed within each party, and is correctly collected, that is: Trapdoor(α) = TrapdoorCollection({Trapdoor(α_i)}).

Also, we have to ensure that the secrecy and security are not compromised at this new scheme. Here, we define the secrecy of DPEKS as follows.

Definition 3 (The Secrecy of DPEKS):

1) For any bank B_i, he should never reveal his secret key share α_i to any other bank B_j, where $j \neq i$.
2) For any bank B_i, he should never learn the secret key share α_j of any other bank B_j through cryptanalysis, where $j \neq i$.

In the following, we will provide the detailed construction and proof of correctness of our DPEKS scheme. The secrecy is proved in security analysis Section VII, where we give rigorous security proofs.

Theorem 1: DPEKS scheme satisfies correctness.

Proof: The first part in Definition 2 is intuitive and is guaranteed in the DKG in Section V-A. The second part, decomposition and collection correctness is the major difference from the original PEKS, and is the core part of DPEKS correctness. This part is similar to the original PEKS scheme, which relies on the bilinear pairing, which is a variant of computational Diffie–Hellman problem (CDH) [25].

We construct the cryptographic model using the Bilinear Maps. Let G_1, G_2 be two groups of prime order p. This indicates that G_1 and G_2 are finite cyclic groups of order p. Let e be a bilinear map where $e : G_1 \times G_1 \rightarrow G_2$. The map e, as a cryptographic bilinear map, also known as a pairing, must satisfy properties of *computability, bilinearity*, and *nondegeneracy* as defined in [26]. Since these definitions are identical with the original paper, we omit them due to the limited pages. Next, we construct two necessary Hash functions $H_1 : \{0,1\}^* \rightarrow G_1$ and $H_2 : G_2 \rightarrow \{0,1\}^{\log_2^p}$ as the *Random Oracles*.

The following steps are the core cryptographic steps and functions of our DPEKS scheme. Assume that we have a transaction information W. When a supervisor S_i requests the banks to generate a list of trapdoors T_w for a list of target illegal keywords w, the aforementioned methods will be executed in sequence.

1) DisKeyGen(n) $\rightarrow$ *(α, β):* Each bank performs the Joint − Feldman's protocol parallelly to generate a list of secret key shares $\alpha_i \in \mathbb{Z}_p^*$, where each bank B_i secretly knows α_i and $\alpha = \sum_{i \in n} \alpha_i$. It also picks a generator g of group G_1. Then, it outputs the public key $\beta = [g, h = g^{\alpha}]$ (broadcasted) and α_i (secretly stored to each). Here, we obtain the asymmetric key pair of all bank.
2) DPEKS(β, W) $\rightarrow$ *C:*
   a) Computes $t = e(H_1(W), h^r) \in G_2$, for a random $r \in \mathbb{Z}_p^*$, where W stands for a vector of to-be encrypted transaction information.

b) Outputs $C = \mathsf{DPEKS}(\beta, W) = [g^r, H_2(t)]$, where C is a vector of encrypted transaction wordlist. This encrypted transaction data is also written in the transaction chain as a history.

3) $\mathsf{TrapdoorLocal}(\alpha_i, w) \rightarrow T_{w_i}$: Outputs a trapdoor list fraction $T_{w_i} = H_1(w)^{\alpha_i} \in G_1$ of the target criminal keywords list w.
4) $\mathsf{TrapdoorCollection}(\{T_{w_i}\}, T_{w_j}) \rightarrow T_w$: Receives all trapdoors T_{w_i} from broadcast, B_j computes the combination of trapdoor list $T_w = \prod_{i \in n \setminus j} T_{w_i} \cdot T_{w_j}$. All banks then broadcasts T_w for cross-verification using PBFT algorithm.
   a) If cross-verification pass, accept, and exit.
   b) Otherwise, discard results.
5) $\mathsf{Test}(\beta, C, T_w) \rightarrow Res$: Let $C = [W_A, W_B]$, test if $H_2(e(T_w, W_A)) = W_B$. If so, output YES; if not, output NO.

The key is to make sure the secret key is linearly shared ($\alpha = \sum_{i \in n} \alpha_i$), and hence the functional output (trapdoor T_{w_i}) can be linearly collected due to group properties. As shown in Section V-A, by generation, the secret key share is guaranteed to sum up to the secret key α ($\alpha = \sum_{i \in n} \alpha_i$). So it is critical that the fractional trapdoors can collect up to the correct, complete trapdoor. This is the major difference between DPEKS and PEKS, and there is no change in the core bilinear map construction, so we will focus proving the correctness of our modified part.

For any finite cyclic group G of p order, it is defined as $G = <g> = \{g^n | n \in \mathbb{Z}, n \leq p\}$, where g is the generator and "$\cdot$" is the group operation. According to the definition, for any $i \in \mathbb{Z}$, $g^{i+1} = g^i \cdot g^1$. Hence, we can know

$$\text{for all } i, j \in \mathbb{Z} \quad g^{i+j} = g^i \cdot g^j \tag{3}$$

which is critical.

The original $\mathsf{Trapdoor}$ function is basically computing the exponential of group operation $H_1(w)^\alpha$, where $H_1(w)$ is an element in the group G. According to the definition of the cyclic group, $H_1(w)$ can be represented as g^x. As x is arbitrary and noncritical, we can simplify it to g. Therefore, according to the linear attribute in (3), we can let each local trapdoor T_{w_i} to be computed by secret key share α_i at local. Then every bank B_i broadcasts T_{w_i}, collects all of them and uses $\mathsf{TrapdoorCollection}$ function to combine them into the final trapdoor T_w

$$\begin{aligned} T_w &= \mathsf{Trapdoor}(\alpha) \\ &= H_1(w)^\alpha \\ &= g^\alpha \\ &= \prod_{i \in n} \mathsf{TrapdoorLocal}(\alpha_i) \\ &= \prod_{i \in n} H_1(w)^{\alpha_i} \\ &= \prod_{i \in n} g^{\alpha_i} \end{aligned} \tag{4}$$

where $\alpha = \sum_{i \in n} \alpha_i$. ■

Theorem 2: DPEKS scheme satisfies secrecy.

Note that although this design is intriguing and delicate, we must guarantee that no secret shares α_i can be inferred or stolen by the adversary. We give security proofs in Section VII, guaranteeing that although we pass the trapdoor shares T_{w_i} to other banks, the secret key share α_i is always kept safe from cryptanalysis.

VI. System Design

In this section, we present the design of NormaChain. We begin by formulating the major design goals, then we specify the core techniques used in a blockchain setting. Lastly, we present the specifications of our proposed NormaChain framework.

A. Design Goal

We conceive the following design goals of NormaChain.

1) Privacy-preserving supervision ability and crime traceability.
2) Autonomous transaction handling.
3) Transaction efficiency and data integrity.

The sharding blockchain network ensures the transaction efficiency and data integrity. The DPEKS scheme along with the digital ID (DID) guarantees the ability of supervision on transactions. The design of E-smart contract (E-contract) ensures autonomous transaction management and enforced execution.

B. Design of the Blockchain System

The three-layer sharding network is supported by the blockchain technology with multiple core components. We specify the design as follows.

1) Digital ID: DID is the universal, unique identifier of one real person or a merchant. All contracts and transactions require the usage of DID to transfer money.

In this paper, we adopt the most popular address format of the DID that is used in the current Ethereum standard, 64 hexadecimal digits.[1] The retrieval of the DID first requires formal registration from the bank. The bank collects the information of the applicant who wishes to obtain a DID and verifies it.

After collecting all the information needed, the bank uses an SHA-256 hash function to generate a 256-bit long string as a DID of this person or merchant. The connection of the DID and Real ID is stored in this bank's local database. In this setting, all banks are responsible for their customers' private information. If an illegal transaction is flagged and results in a search warrant, local databases will be used to inform authorities on the criminal's true identity. The DID data structure is shown in Fig. 3.

2) Token: The settlement method in almost all blockchain networks is done by the tokens or cryptocurrencies. It is a replacement of money that is transferred *within* the network. This is because the frequent transfer of the fiat currency is canceling out the convenience and fast response advantages of the blockchain network. Here, we denote the NorMaCoin as NMC token for the ease of settlement within the system. All balance of users are stored in their DID address as NMC tokens, and must be bought from their corresponding banks as initial

[1] It used to be 40 hexadecimal digits. The Ethereum updated this in v0.4.17.

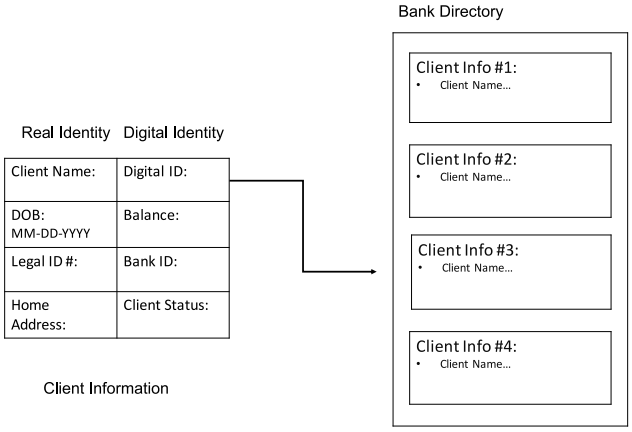


Fig. 3. DID data structure.

Algorithm 1 PBFT-Based Bank Identity Verification Algorithm

Require: B_i: The bank who needs to be verified; B_j: any other qualified bank; $|B|$: the number of all banks; R: verification result.

1: B_i: Broadcasts the request of authentication REQ_{AUTH} and B_i's digital certificate $CERT_{B_i}$ to all
2: B_j: Verifies $CERT_{B_i}$ and broadcasts its decision
3: B_j: Receives all replies from B_k $(k \neq j)$
4: **if** The number of same decision R is more than $\frac{|B|-1}{3}+1$, **then**
5: return R
6: **else**
7: return *NULL*
8: **end if**

balance. After that, it can be transferred upon transaction. All the customers and merchants can settle transactions using this token balance, but only when any user intends to quit this system, he can cash-in his tokens into fiat currencies from his bank. This design principle of lowering external transactions (fiat currency settlement) by increasing internal transactions (token) is very similar to the recent lightning network optimization of the Bitcoin [27].

3) Consensus Algorithm: Consensus algorithms unify different opinions of verifiers, and return a final decision agreed by all. In this paper, we couple X.509 digital certificate verification standard [28] with the PBFT [17] to verify the identity of B_i. Then the B_i alone verifies the financial transaction. As a result, the computational liability is minimized while also maintaining the decentralization of the system. It is important to note, however, that PBFT requires no more than 1/3 malicious nodes in the network. While this may be thwarted by increasing the total number of verifiers, the resulting high communication cost can degrade the algorithm's efficiency.

The PBFT-based verification algorithm of the bank B_i is shown in Algorithm 1.

All bank nodes are interconnected and therefore are able to obtain other bank's information. This decentralized topological structure prevents a single-node failure effectively. It was

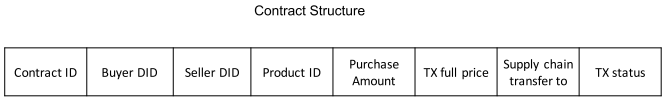


Fig. 4. E-commerce contract data structure.

proven in [17] to be able to work even with up to $([1 + |B|]/3)$ malicious or faulty nodes. The only drawback is high cost of interconnected communication between the nodes. It is easy to see that as we need pairwise, or full communication, we have a communication overhead of $O(n^2)$. Luu *et al.* [19] demonstrated in experiments that the communication cost is only considerably high when the approval nodes exceed 100. However, in this paper, since the approval nodes represent established banks, the quantity of nodes should remain low.

C. E-Contracts

The E-commerce contracts represent a digital transaction showing *who* is buying *what* from *whom*, and paying *how much*. This idea is first brought up by the Ethereum community and they invented a standardized framework out of it. Since Ethereum framework poses a high overhead on lightweight devices, we develop our own lightweight E-contract payment platform using C++.

When a buyer wishes to initiate a transaction, he requests it by messaging the seller (step 1 in Fig. 2). Then both buyer and seller client would initiate and generate an identical contract for one transaction and send it to the approval layer simultaneously (step 2 in Fig. 2). The structure of the E-contracts is shown in Fig. 4. This design is to make sure that no buyer or seller alone can tamper the contract unilaterally. Once the contract is approved by the bank, it is automatically sent to the supply chain for further processing (step 3 in Fig. 2). The contract generation and approval diagram are shown in Fig. 2.

D. Design of Transaction and Supervision Chains

The sharding network divides itself into different layers. As a result, each layer should also have its respective chain to record a specific kind of history. In this paper, we have a transaction chain, which is shared by the transaction and approval layer to record transaction histories and a supervision chain for recording all attempts to search and supervise. Overall, these chains ensure that the transactions are all recorded with integrity and nonrepudiability while inhibiting abuse of power by supervisors.

The transaction and supervision chains are all designed using the well-known *Merkle trees* [29] for fast verification purposes. For simplicity concerns, every approved transaction (recorded as contract) is encrypted with the public key β and recorded as an individual block in the chain. It leaves a possibility to easily upgrade and scale in the future. A simplified block structure is shown in Figs. 5 and 6.

E. Distributed Ledger

The distributed ledger locally stores the necessary public information of this node to carry out basic blockchain tasks.

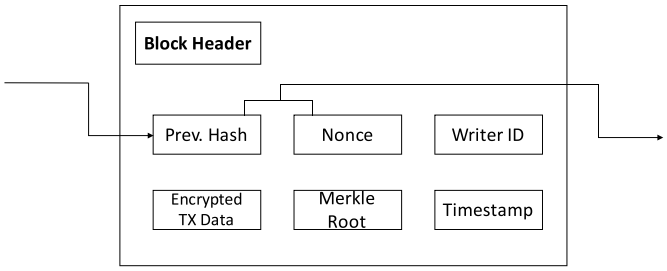


Fig. 5. Transaction chain structure (simplified).

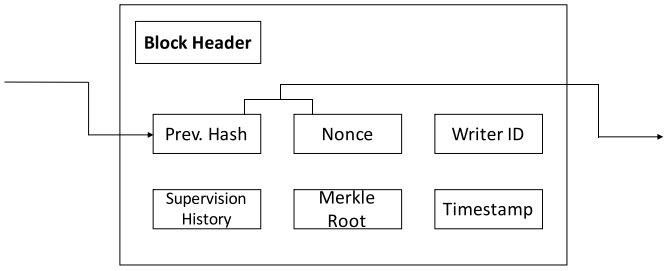


Fig. 6. Supervision chain structure (simplified).

As a result, each role in the network has its own distributed ledger construction.

1) User Node (Customer/Merchant): As we do not require ordinary users to mine the transaction, although they still have the freedom to access the transaction chain, the chain itself is not mandatory for them to store. All customers or merchants, however, must locally store their DID and their corresponding private key.

2) Bank Node: Banks have the responsibility of approving the transaction contracts and writing their encrypted contracts to the transaction chain. Therefore, banks will store the whole transaction chain, their local customer database, and the public key in order to carry out their tasks.

3) Supervision Node: The supervisors are able to search along the transaction chain using their required target criminal keyword list w but must also write their activity onto the supervision chain. Additionally, it is required that each supervision node stores the whole supervision chain to protect the data integrity.

F. System Framework

In this section, we provide our specifications of NormaChain framework. It automatically collects, verifies and records all the transactions, and also carries out supervisions on it. The specific steps of our framework are shown as follows.

1) The system generates the key pair $(\beta, \{\alpha_i\}) = \mathsf{DisKeyGen}(\mathsf{n})$. Then broadcasts the public key β to all nodes.
2) Any buyer U_i who wishes to initiate transactions sends a message msg to the seller U_j, and both of them generate an identical transaction contract and send to the buyer's bank B_i.
3) After the PBFT verification of the bank B_i's identity, B_i verifies the contracts, decides the contracts to be Approved or Declined, and then writes on the transaction chain by first fracturing the transaction contract into words W, and encrypting it as $C = \text{PEKS}(\beta, W)$. The contract is then sent to the supply chain for next step logistics.
4) If any supervisor S_i wishes to search for a target criminal keyword list w, he sends w to all banks B for a consensus process. If this target criminal keyword list is unreasonable, and $2/3$ of all banks' approval is not achieved, then trapdoor computation on this keyword list is rejected. Otherwise, B as a whole returns a corresponding trapdoor list $T_w = \mathsf{TrapdoorCollection}(\mathsf{TrapdoorLocal}(\alpha_i, w))$.
5) The supervisor S_i can test the recently added, encrypted transaction C with $\text{Res} = \text{Test}(\beta, C, T_w)$. S_i finally receives a list of $\{0,1\}^{|T_w|}$ indicating the existence of the target criminal keyword. This result is then written in the supervision chain and if legally warranted, S_i can decrypt and look into that block by requesting the construction of α. Note that this is the only exception to reconstruct α explicitly.
6) If there is a security threat, or explicit leakage on the secret key α, the system revokes keypair, go to stage 1); Otherwise go to stage 2).

VII. Security Analysis

In this section, we analyze the security features of NormaChain.

A. Adversarial Supervisor

1) Proof of Security Against CCA: The supervisor's adversarial model has been introduced in the previous section, which is honest but curious which means the supervisor would perform his duties but stays curious about full customer's plaintext information. According to our design goals, we must preserve a benign user's privacy; that is, the supervisors should not know anything other than the $\{0, 1\}$ result of his detection of criminal keywords. We now prove that our NormaChain system preserves customer privacy from supervision.

CCA is a widely used security evaluation to measure the adversary's advantage on guessing the answer. It usually refers to an attack where the adversary (in our scenario, supervisor $\mathcal{A}_{S_i}$) is requesting the encrypter (banks) an adequate amount of ciphertexts (trapdoors) until they can have a higher advantage of guessing the correlation of the keyword and the trapdoor. In our scheme, we prohibit the supervisor to know anything other than the $\{0, 1\}$ when using their trapdoors.

We roughly perceive that if a PEKS+blockchain scheme is adaptively CCA-secure, then this scheme is customer privacy preserving against the supervisor. In NormaChain, we constructed our PEKS scheme using the bilinear map and the CDH Problem. The bilinear map construction of PEKS scheme is rigorously proven in the [2] to be semantically secure against adaptive CCA (CCA2 secure). As we are constructing our DPEKS scheme according to this exact mathematical core, we claim to have CCA2-security and can, therefore, protect customer privacy from curious supervisors.

B. Adversarial Bank

The adversary model of banks was defined similarly as the supervisors. They are also honest but curious but this time, they can be curious on recovering the complete secret key, such that one can decrypt the encrypted information, or compute any trapdoor he wants. There are two ways of actually achieving this: by cryptanalysis and collusion. Recall in Section IV, Theorem 2 claims that DPEKS satisfies secrecy. We now prove both of these attacks are intractable.

1) Proof of Security Against Cryptanalysis: As we mentioned earlier, the computation of the complete trapdoor T_w is by a multiparty computation. So for each adversary bank $\mathcal{A}_{B_i}$, the information of other bank's secret key share is only accessible from the received bank's trapdoor fractions. We now prove that for any bank B_i, he cannot recover the any bank's secret key share B_j from his fractional trapdoor T_{w_j}.

The proof relies on the hardness of the discrete logarithm problem. Here, we provide the definition of the group discrete logarithm problem.

Definition 4: For a finite cyclic group G of p order, G is defined as $G = <g> = \{g^n | n \in \mathbb{Z}, n \leq p\}$, where g is the generator and the "$\cdot$" multiplication is the group operation. Let a be an element of G. An integer k that solves the equation $b^k = a$ is noted as a discrete logarithm problem of a to the base b. Or simply to compute $k = \log_b^a$.

And also one of the foundational hardness theorems of modern cryptography, that is discrete log problem hardness theorem.

Theorem 3: For any sufficiently large group order p, the group discrete logarithm problem is computationally intractable.

As mentioned in Section V, the trapdoor $T_{w_j} = g^{\alpha_j}$. As a result, for any adversary $\mathcal{A}$, the goal of computing the secret key share of bank B_j is to compute $\alpha_j = \log_g^{T_{w_j}}$. As stated in Theorem 3, this problem is computationally intractable. Hence, we can guarantee the security against cryptanalysis on public information.

2) Proof of Security Against Collusion: Another way of stealing the full secret key is to secretly collude and collect all pieces of secret key share. That is: assuming all banks are semi-honest and each has a possibility of colluding. An adversary $\mathcal{A}$ (could be a bank or supervisor) attempts to corrupt and take control of all n banks. Assume each bank's possibility of collusion follows a discrete normal distribution $X \sim N(\mu, \sigma^2)$, which we set the average probability of collision is 0.5 ($\mu = 1/2$), and the standard deviation σ is $1/4$. For each round, the possibility of each bank choosing to collude is sampled as p_i, and $1 - p_i$ otherwise. So the probability of collision for each round is $p = \prod_{i \in n} p_i$. By simulated sampling of MATLAB, when $n = 10$, the typical value of p is approximately $1e^{-4} \sim 1e^{-3}$. When $n = 50$, the typical value of p is approximately $1e^{-16} \sim 1e^{-15}$. When $n = 100$, the typical value of p is approximately $1e^{-32} \sim 1e^{-30}$, which is statistically satisfying.

C. Adversarial User

Since the users in our system are bare minimal responsibilities for maintaining the transaction chain, the user adversarial model does not have much to attack against. The adversarial user model in this paper is set to have a simple goal, which is to tamper the transaction contract and his balance [22].

1) Proof of Security Against Tampering With Data: As stated in the design of our E-contracts, when users want to initiate a transaction, both the buyer and seller generate an identical E-contract and send it to the buyer's bank for approval. With this design, assuming the communication channel is safe (using SSL/TLS), then the unilateral tampering of the transaction contract can never succeed. Also, the balance of the user is also traded in the format of an E-contract and blockchain, so the balance data integrity of the node is also guaranteed.

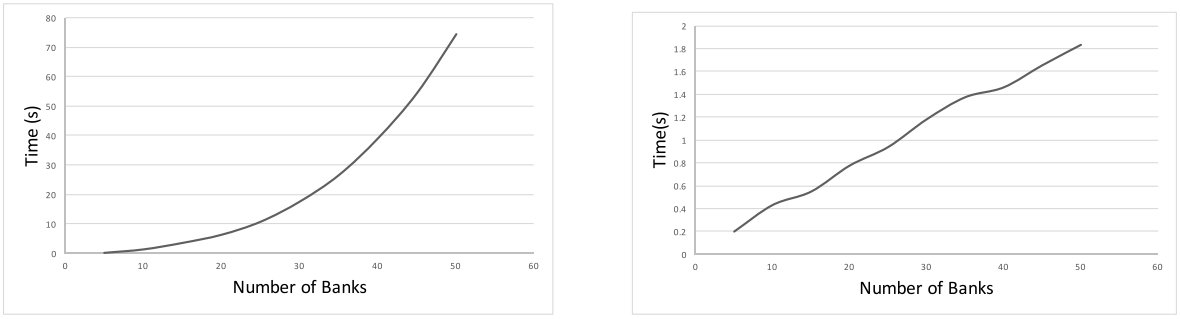
VIII. EVALUATION

In this section, we present the evaluation of NormaChain based on our self-implemented blockchain infrastructure. Our evaluation testifies NormaChain in three major aspects, the efficiencies of transaction/supervision, i.e., the key distribution/trapdoor collection latencies, the transaction/supervision latencies, and the accuracy of supervision.

A. Evaluation Setup

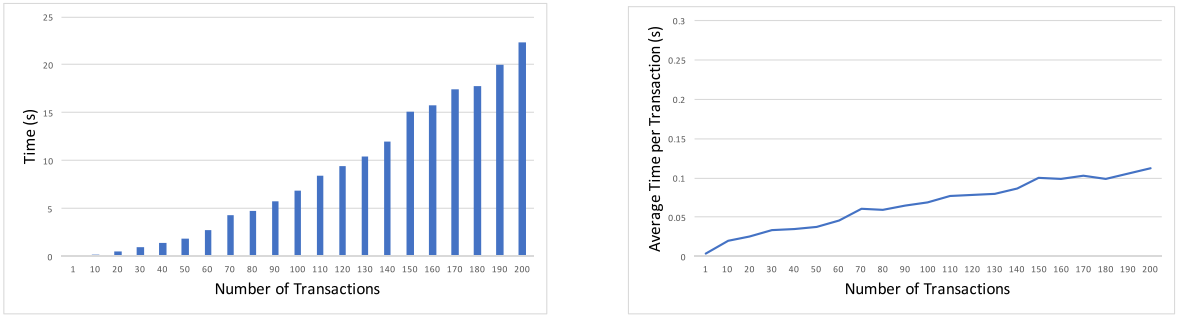
For evaluation purpose, we fully implemented the NormaChain infrastructure that can specifically fit the application setting of IoT devices, which have limited computational powers and weak security strengths. Even though there are well-known blockchain-based programming frameworks in the industry, such as Ethereum [30], they all create a virtual machine upon which the developers implement codes. Then the codes are compiled with memory-safe language compilers or just-in-time compilers and executed under the protection of the virtual machine. Even though this configuration can guarantee maximized protection in order to defend against memory corruption attacks, it sacrifices the efficiencies. According to the demonstrations presented in two recent works [31], [32], the authors of both papers ran Ethereum on a Raspberry Pi, an embedded device running on a 1.4 GHz 64/32-bit quad-core ARM Cortex-A53 CPU, which assembles most IoT devices on the hardware level, and found that the overhead can reach up to 10 min for one single transaction. Having realized this fact, we implement the prototype of NormaChain using native languages, i.e., C/C++, in order to pursue better performance. As a result, we write 3601 lines of codes in C/C++ to establish the prototype of NormaChain. Then we configure the project based on CMake [33], a cross-platform building tool so that our project can be deployed in almost any platforms including Android, iOS, Linux, Windows, and Mac OS easily.[2] Finally, we evaluate our tested prototype on two Raspberry Pis with ARM instruction set which assembles most IoT devices and mobile devices in the industrial market, and a laptop running on Ubuntu 16.04 LTE with Intel Core i7-6500U CPU 2.50GHz $\times$ 4 CPU. And for the following context, our evaluations are conducted with one buyer, one seller, three approvers

[2]The source codes of our implementation are available online. [Online]. Available: https://github.com/yinhaoxiao/NormaChain



(a) (b)

Fig. 7. Setup efficiency. (a) Time consumption of DKG versus number of banks. (b) Time consumption of trapdoor collection versus number of banks.



(a) (b)

Fig. 8. Transaction efficiency on laptop. (a) Time consumption versus number of transactions. (b) Averaged time consumption versus number of transactions.

(regular banks), 5–50 banks for the key distribution/trapdoor collection efficiency and one supervisor for other two aspects.

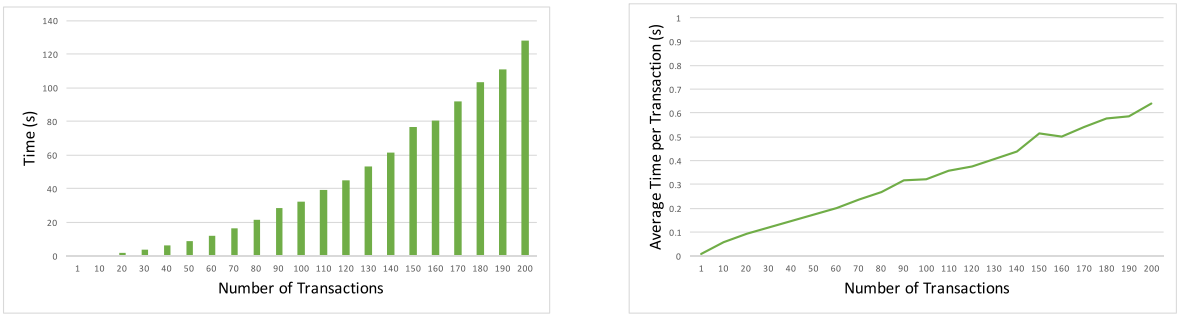
B. Key Distribution and Trapdoor Collection Efficiency

As a part of DPEKS setup, we conduct two tests to evaluate the efficiency: 1) time for DKG and 2) time for trapdoor collection. In the first experiment, timestamps were recorded in an effort to examine the number of seconds needed for private key distribution across n banks. Similarly, the second experiment sought to determine the time consumption when collecting trapdoors from n banks. In this case, the number of banks tested ranged from 5 to 50 with an interval of 5. Our results from the first test showed 5 and 50 banks taking 0.19 and 74.23 s, respectively. In contrast, our second test only required an additional 0.2 s for every five additional banks. These results match our expectations as the local test machine could not efficiently handle multithreaded key distribution. With this in mind, we expect multiple computationally capable machines to drastically enhance key distribution efficiency. The data for the results are shown in Fig. 7.

C. Transaction/Supervision Efficiency

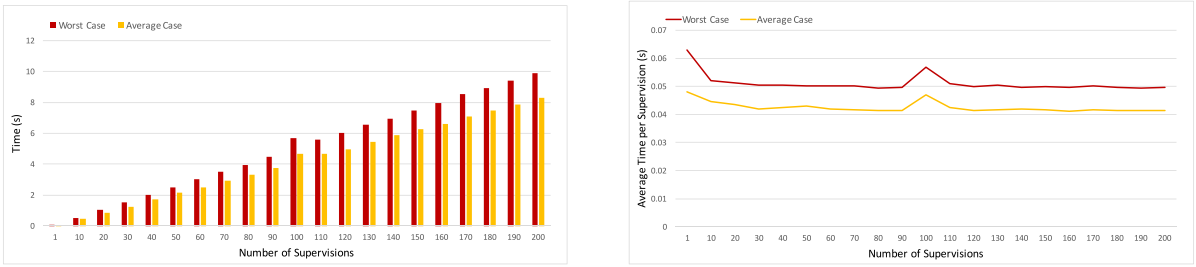
We evaluated the efficiencies of both conducting a transaction and a keyword search. For evaluating the efficiency of transactions, we measure the time elapsed for transactions running in a *single thread*. We conducted two sets of experiments: one with the seller and buyer running on the Raspberry Pis, and the other one with the seller and buyer running on the laptop. As a result, the time elapsed for one transaction is 0.003466 s for laptop environment and the time elapsed for one transaction for Raspberry Pi environment is 0.008796 s. Similarly, the numbers for 10 transactions, 50 transactions, 100 transactions, and 200 transactions are 1.00099/0.572394 s, 0.512963/8.6764 s, 6.90184/32.301 s, and 22.3544/127.718 s, respectively, (with the format laptop/Raspberry). In other words, despite running on a lightweight laptop with only a *single thread*, our implementation is able to complete 288.52 transactions per second (TPS). And running on computationally limited embedded IoT devices such as Raspberry Pis, we can achieve 113.69 TPS. This is a very convincing experiment showing M2M trading model autonomy of IoT devices. The detailed results are shown in Figs. 8 and 9.

For evaluating the efficiency of supervision, we conducted two sets of experiments. Both experiments were conducted in the laptop environment. In the first set of experiments, the keywords sent from the supervisor are not presented in the prestored contracts. In this case, the banks have to perform the worst-case search, i.e., searching all the contracts. In the second set of experiments, the keywords for searching exist in the prestored contracts. We evaluate these two sets of experiments based on the situations where there are 1 prestored contract, 10 contracts, 20 contracts and all the way to 200 with an increment of 10 contracts each time. As a result, for the situation where there is only 1 contract, a keyword search can be completed in 0.062977 s if the keyword does not exist in the contract; and it can be completed in 0.048097 s if the



(a) (b)

Fig. 9. Transaction efficiency on Raspberry Pis. (a) Time consumption versus number of transactions on Raspberry Pis. (b) Averaged time consumption versus number of transactions on Raspberry Pis.



(a) (b)

Fig. 10. Supervision efficiency. (a) Time consumption versus number of Supervisions. (b) Averaged time consumption versus number of Supervisions.

TABLE I
EXAMPLE OF A SUSPICIOUS TRANSACTION

Message	Cat	buys	cocaine	from	Dr.	drugdealer
Target	cocaine			drugdealer		
Ground Truth	0	0	1	0	0	1
Test Result	0	0	1	0	0	1
Accuracy	100%					

TABLE II
EXAMPLE OF ANOTHER SUSPICIOUS TRANSACTION

Message	Cat	buys	ice	from	Dr.	juggler
Target	cocaine			drugdealer		
Ground Truth	0	0	1	0	0	1
Test Result	0	0	0	0	0	0
Accuracy	0%					

keyword exists in the contract. Similarly, such numbers for 10 contracts are 0.520825 and 0.446612 s; the numbers for 50 contracts are 2.5074 and 2.14615 s; the numbers for 100 contracts are 5.69439 and 4.68922 s, and the numbers for 200 contracts are 9.91123 and 8.28285 s. The detailed results are shown in Fig. 10.

D. Supervision Accuracy

For evaluating the accuracy of supervision, i.e., testing the accuracies of our implementation based on the DPEKS algorithm, we evaluate our framework from three aspects, i.e., calculating the overall accuracy rate, calculating the false positive rate, and calculating the false negative rate. We leverage the same dataset used for evaluating the efficiency of supervision mentioned above for this evaluation. As a result, as shown in Table I, the overall accuracies of all the searches are 100%, with 0% false positive rate and 0% false negative rate. It is worth mentioning that even though the accuracy of supervision in NormaChain can reach 100% given all illegal keywords, it does not rule out the possibilities of crimes due to semantics restrictions. For example, Table II shows an example where the drug dealers use jargons instead of standard words to complete a transaction. In this case, the supervisor searches keywords "cocaine" and "drug dealer" to uncover potential drug transaction, however, the criminals use the jargons "ice" and "juggler" to circumvent the supervision. Therefore, in the future study, we intend to study the possible techniques to strengthen our infrastructures, such as machine learning or NLP for recognizing black keywords demonstrated in the work of Yang *et al.* [34].

IX. CONCLUSION

In this paper, we propose NormaChain, a normalized autonomous transaction settlement system for IoT-based E-commerce. Our main contribution in this paper is that we first propose a three-layer sharding blockchain network model. Then we design an innovative decentralized PEKS cryptosystem and prove it is secure against CCA, cryptanalysis, and collusion. And finally, we deliver a full autonomous transaction settlement system with a C++ implemented efficient prototype. Experiments show that our system is not only efficient on transaction handling, and is also accurate on privacy-preserving illegal criminal keyword search.

REFERENCES

[1] J. Martin, "Lost on the silk road: Online drug distribution and the 'cryptomarket,"' *Criminol. Crim. Justice*, vol. 14, no. 3, pp. 351–367, 2014.

[2] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.*, 2004, pp. 506–522.

[3] J. Huang, C.-C. Xing, S. Shin, F. Hou, and C.-H. Hsu, "Optimizing M2M communications and quality of services in the IoT for sustainable smart cities," *IEEE Trans. Sustain. Comput.*, vol. 3, no. 1, pp. 4–15, Jan./Mar. 2017.

[4] P. Janson *et al.*, *Electronic Payment Systems*, IEEE Computer, 1996.

[5] N. Asokan, "Fairness in electronic commerce," Ph.D. dissertation, Dept. Comput. Sci., Univ. Waterloo, Waterloo, ON, Canada, 1998.

[6] B. Meng and Q. Xiong, "Research on electronic payment model," in *Proc. 8th Int. Conf. Comput. Support. Cooper. Work Design*, vol. 1, 2004, pp. 597–602.

[7] J. S. Kiernan. (2017). *Credit Card and Debit Card Fraud Statistics*. [Online]. Available: https://wallethub.com/edu/credit-debit-card-fraud-statistics/25725

[8] A. G. González, "PayPal: The legal status of C2C payment systems," *Comput. Law Security Rev.*, vol. 20, no. 4, pp. 293–299, 2004.

[9] J. Gao, K. Edunuru, J. Cai, and S. P. D. Shim, "P2P-Paid: A peer-to-peer wireless payment system," in *Proc. 2nd IEEE Int. Workshop Mobile Commerce Services (WMCS)*, 2005, pp. 102–111.

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

[11] S. Roos, P. Moreno-Sanchez, A. Kate, and I. Goldberg, "Settling payments fast and private: Efficient decentralized routing for path-based transactions," *arXiv preprint arXiv:1709.05748*, 2017.

[12] R. Pass and E. Shi, "Thunderella: Blockchains with optimistic instant confirmation," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.*, 2018, pp. 3–33.

[13] E. Kokoris-Kogias *et al.*, "OmniLedger: A secure, scale-out, decentralized ledger via sharding," in *Proc. IEEE Symp. Security Privacy (SP)*, 2018, pp. 583–598.

[14] F. Tian, "An agri-food supply chain traceability system for China based on RFID & blockchain technology," in *Proc. IEEE 13th Int. Conf. Service Syst. Service Manag. (ICSSSM)*, 2016, pp. 1–6.

[15] A. Xu *et al.*, "A blockchain based micro payment system for smart devices," *Signature*, vol. 256, no. 4936, p. 115, 2016.

[16] P.-W. Chen, B.-S. Jiang, and C.-H. Wang, "Blockchain-based payment collection supervision system using pervasive bitcoin digital wallet," in *Proc. IEEE Wireless Mobile Comput. Netw. Commun. (WiMob)*, 2017, pp. 139–146.

[17] M. Castro and B. Liskov, "Practical Byzantine fault tolerance," in *Proc. OSDI*, vol. 99, 1999, pp. 173–186.

[18] D. Schwartz *et al.*, "The ripple protocol consensus algorithm," San Francisco, CA, USA, Ripple Labs Inc., White Paper, vol. 5, 2014.

[19] L. Luu *et al.*, "A secure sharding protocol for open blockchains," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2016, pp. 17–30.

[20] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proc. IEEE Symp. Security Privacy (SP)*, 2000, pp. 44–55.

[21] H. Abu-Libdeh, R. Van Renesse, and Y. Vigfusson, "Leveraging sharding in the design of scalable replication protocols," in *Proc. 4th Annu. Symp. Cloud Comput.*, 2013, p. 12.

[22] R. Wang, S. Chen, X. Wang, and S. Qadeer, "How to shop for free online—Security analysis of cashier-as-a-service based Web stores," in *Proc. IEEE Symp. Security Privacy (SP)*, 2011, pp. 465–480.

[23] T. P. Pedersen, "A threshold cryptosystem without a trusted party," in *Proc. Workshop Theory Appl. Cryptograph. Techn.*, 1991, pp. 522–526.

[24] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Secure distributed key generation for discrete-log based cryptosystems," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.*, 1999, pp. 295–310.

[25] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Proc. Annu. Int. Cryptol. Conf.*, 2001, pp. 213–229.

[26] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, "Aggregate and verifiably encrypted signatures from bilinear maps," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.*, 2003, pp. 416–432.

[27] J. Poon and T. Dryja. (2016). *The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments, DRAFT Version 0.5.9.2*. [Online]. Available: https://lightning.network/lightning-network-paper.pdf

[28] R. Housley, W. Ford, W. Polk, and D. Solo, "Internet x. 509 public key infrastructure certificate and CRL profile," IETF, Fremont, CA, USA, Rep. RFC 2459, 1998.

[29] M. Szydlo, "Merkle tree traversal in log space and time," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.*, 2004, pp. 541–554.

[30] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," Ethereum Project, Yellow Paper, vol. 151, pp. 1–32, 2014.

[31] Y. Zhang, S. Kasahara, Y. Shen, X. Jiang, and J. Wan, "Smart contract-based access control for the Internet of Things," *arXiv preprint arXiv:1802.04410*, 2018.

[32] A. Ouaddah, A. A. Elkalam, and A. A. Ouahman, "FairAccess: A new blockchain-based access control framework for the Internet of Things," *Security Commun. Netw.*, vol. 9, no. 18, pp. 5943–5964, 2016.

[33] K. Martin and B. Hoffman, *Mastering CMake: A Cross-Platform Build System*. Clifton Park, NY, USA: Kitware, 2010.

[34] H. Yang *et al.*, "How to learn Klingon without a dictionary: Detection and measurement of black keywords used by the underground economy," in *Proc. IEEE Symp. Security Privacy (SP)*, May 2017, pp. 751–769.

Chunchi Liu received the bachelor's degree (with distinction) in science from Beijing Normal University, Beijing, China. He is currently pursuing the Ph.D. degree at the Computer Science Department, George Washington University, Washington, DC, USA.

He is currently with Prof. Xiuzhen (Susan) Cheng's Group, George Washington University. He has had several technical papers published in top international conferences and journals, such as IEEE INFOCOM and the *International Journal of Sensor Networks*. His current research interests include blockchain, cryptocurrency, security, and applied cryptography.

Yinhao Xiao received the bachelor's degree in information and computing science from the Guangdong University of Technology, Guangzhou, China, in 2012, and the master's degree in applied mathematics and second master's degree in computer science from George Washington University, Washington, DC, USA, in 2014 and 2015, respectively, where he is currently pursuing the Ph.D. degree at the Department of Computer Science.

His current research interests include a broad area of security and privacy protection, especially system security (more specifically, mobile security and IoT security) and social network privacy.

Vishesh Javangula is currently pursuing the Bachelor of Science degree in biology (with a minor in computer science) from George Washington University, Washington, DC, USA.

He is currently an Undergraduate Research Assistant with Prof. Xiuzhen (Susan) Cheng's Group, George Washington University, focusing on blockchain applications for use in industry settings.

Qin Hu received the M.S. degree in computer science and B.S. degree in electronic engineering from Beijing Normal University, Beijing, China, in 2017 and 2014, respectively. She is currently pursuing the Ph.D. degree in computer science at George Washington University, Washington, DC, USA.

Her current research interests include cognitive radio networks, wireless network, and game theory.

Shengling Wang (M'15) received the Ph.D. degree from Xi'an Jiaotong University, Xi'an, China, in 2008.

She is currently an Associate Professor with the College of Information Science and Technology, Beijing Normal University, Beijing, China. She was a Post-Doctoral Researcher with the Department of Computer Science and Technology, Tsinghua University, Beijing. She was an Assistant and an Associate Professor with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, from 2010 to 2013, respectively. Her current research interests include mobile/wireless networks, game theory, and crowdsourcing.

Xiuzhen Cheng (M'03–SM'12–F'15) received the M.S. and Ph.D. degrees in computer science from the University of Minnesota–Twin Cities, Minneapolis, MN, USA, in 2000 and 2002, respectively.

She is a Professor with the Department of Computer Science, George Washington University, Washington, DC, USA. She was a Program Director for the U.S. National Science Foundation in 2006 (full time) for six months and from 2008 to 2010 (part time). She has authored or co-authored over 200 peer-reviewed papers. Her current research interests include privacy-aware computing, wireless and mobile security, smart cyber-physical systems, mobile handset networking systems (mobile health and safety), and algorithm design and analysis.

Dr. Cheng is the founder and the Steering Committee Chair of the International Conference on Wireless Algorithms, Systems, and Applications (WASA) in 2006 and the co-founder of the IEEE Symposium on Privacy-Aware Computing in 2017. She has served on the Editorial Boards of several technical journals, such as the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS and *IEEE Wireless Communications* and the Technical Program Committees of various professional conferences/workshops, such as ACM Mobihoc, ACM Mobisys, IEEE INFOCOM, IEEE ICDCS, IEEE ICC, and IEEE/ACM IWQoS. She has also chaired several international conferences, including IEEE CNS and WASA.