Hybridchain: A Novel Architecture for Confidentiality-Preserving and Performant Permissioned Blockchain Using Trusted Execution Environment

YONG WANG, JUNE LI, SIYU ZHAO, AND FAJIANG YU

Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan 430072, China

Corresponding author: June Li (jeli@whu.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant 51977155.

ABSTRACT

Blockchain is making headlines due to it promises to provide a decentralized, transparent, tamper-resistant, traceable and verifiable historical transaction records that can resist faults of any single node. According to the latest data from State of the Dapps, developers have currently released 3,717 Decentralized Applications (DApps), only three have an average of more than 10,000 daily active users. Most of the real-world DApps exercise little of their potential power. The key reason is that the current permissioned blockchain systems suffer from poor performance and lack of confidentiality. To address this issue, we present Hybridchain, a system that combines blockchain with Trusted Execution Environment (TEE). Hybridchain decouples computation from consensus and adopts hierarchical network to minimize the computational burden and latency of on-chain execution by performing most of the heavy-weight computation off-chain. Hybridchain leverages secure communication protocols to enable each participant to share transaction data in a secure way. To mitigate the small enclave memory restriction of TEE, Hybridchain extends the enclave memory that allows blockchain applications running in TEE to securely store transaction records to the whole key-value storage codes placed outside of TEE. Analysis and experiments of sealed-bid auction show that Hybridchain can support confidentiality-preserving along with high performance.

INDEX TERMS

Permissioned blockchain, trusted execution environment, high performance, hierarchical network, confidentiality, memory extension.

I. INTRODUCTION

Blockchain has recently attracted widespread interest and significant investment of stakeholders across a wide span of industries: from digital currency and financial, to copyright protection, energy transactions, sharing economy and e-government [1]. As a type of blockchain, permissioned blockchain is becoming a key enabling platform for many companies to design and deploy various Decentralized Applications (DApps) and process a large number of digital transactions [2]. In order to provide customers with high Quality of Service (QoS), the business in these systems and applications need to be fast, low-latency, robust, and secure.

The associate editor coordinating the review of this manuscript and approving it for publication was Kaitai Liang.

However, since all transaction data is replicated on all nodes in blockchain network, there is no confidentiality guarantee for transaction data. Each node in blockchain network has access to each byte of smart contract’s data. Statistical methods such as data mining and sociological mining have caused users’ privacy to face major threats [3]. Blockchain performs computation tasks through smart contract on the chain, the block size, the number of instructions and block confirmation time of consensus algorithm strictly limit the time and space resource available for smart contracts [4]. From the perspective of framework, most blockchains adopt the order-execute architecture [5], where nodes package transactions in order and then synchronize them to each node via broadcast, and each node executes the transactions in order. That is, “order” first, then “execute”.
This architecture is difficult to scale. First of all, the execution of transactions in order by all nodes will limit performance of blockchain system. Although concurrent execution of unrelated operations can improve performance, it is hard to achieve concurrency for smart contracts owing to the dependencies between codes are difficult to determine. In addition, the biggest limitation of order-execute architecture is that the transactions executed by all nodes must be deterministic. Due to transactions are required to execute on every full node in the entire network, there are a lot of redundant computation. The performance of entire blockchain network will not exceed a single node. The current permissioned blockchain systems suffer from poor performance and lack of confidentiality.

A. MOTIVATION

Many researchers try to explore solutions to these challenges, such as Multi-Party Computation (MPC), Zero-Knowledge Proof (ZKP) and ring signature technology. Gao et al. [6] proposed BFR-MPC, a multi-party computing scheme based on blockchain to address the fairness and robustness issues in MPC. Bowe et al. [7] performed ZKP to execute offline computation and subsequently produces publicly-verifiable transactions that attest to the correctness on-chain system. Li et al. [8] proposed a ring signature privacy data storage protocol based on Elliptic Curve Cryptography (ECC). The protocol took advantage of the complete anonymity of ring signatures to ensure the security of transaction data and users’ identity privacy in blockchain applications. However, these cryptographic solutions are cumbersome and suffer from heavy performance overhead owing to the inefficient, complex computation protocol and frequent interaction. Note that MPC with more than two parties cannot be deployed due to message complexity overhead.

Another fundamentally different approach to achieve confidentiality is Trusted Execution Environment (TEE). TEE has complementary properties with blockchain [9]. TEE has little overhead and can provide verifiable computation with confidential state via Intel Attestation service (IAS) [10], whereas blockchain nodes generally have limited computation capability and need to expose their entire states for public verification [11]. TEE cannot guarantee availability or reliable access to the network, whereas blockchain can guarantee strong availability and persistence of its state [10]. The combination of blockchain and TEE makes it possible to enhance the performance and confidentiality of blockchain. TEE can be used to provide an underlying trusted network for blockchain, so complex on-chain computation tasks can be performed off-chain and blockchain is used for bookkeeping [12]. TEE can be leveraged to validate the transaction and endorse the validity instead of all the parties validating the transactions. As a secure sandbox implemented by trusted hardware, TEE can ensure that runtime internal data is invisible to the Rich Execution Environment (REE) [13]. Only the interface defined in the code can operate on it. TEE can guarantee the integrity for code and integrity and confidentiality for data [14].

In this paper, we present a new hybrid blockchain architecture called Hybridchain through the synthesis of TEE and blockchain. Hybridchain enables the secure movement of blockchain processing from on-chain to off-chain dedicated computation resources. The off-chain nodes execute the transaction proposals and corresponding smart contracts inside the TEE, while associated on-chain nodes validate the execution results and maintain the distributed ledger. By combining the off-chain computation and on-chain verification, enterprises and individuals can deploy the data processing model on-chain. The private data is processed off-chain, while the final verifiable results are stored on-chain. We verified the feasibility of Hybridchain through sealed-bid auction use case. Analysis and experiments show that Hybridchain can support confidentiality-preserving along with high performance.

B. OUR CONTRIBUTIONS

The contributions of this paper can be summarized as follows:

- **Hybrid architecture.** We propose Hybridchain, a new architecture that enhances confidentiality and performance of permissioned blockchain by combining blockchains with TEE.

- **Hierarchical network.** We design a hierarchical network structure to decouple the execution of smart contracts from the on-chain consensus mechanism, and put the initial state and final state on-chain instead of every state. We also design secure communication protocols to enable each participant to share transaction data in a secure way.

- **Memory extension of TEE.** To mitigate the small enclave memory restriction of TEE, we design a new secure key-value memory system that allows blockchain applications running in TEE to securely store transaction records to the whole key-value storage codes placed outside of TEE.

C. ORGANIZATION

The rest of this paper is organized as follows: In section II, we introduce and discuss the related work. In section III, we briefly introduce blockchain and TEE and explain some important features of them. In section IV, we present the overall system architecture, design the hierarchical network, and expand the memory of TEE. In section V, we analyze the security and privacy guarantees provided by our system. In Section VI, we provide the experimental evaluation of the proposed system. Finally, we conclude by the future work discussion in Section VII.

II. RELATED WORK

TEE is on the path to play a key role in blockchain system, especially for permissioned blockchains and Decentralized Applications (DApps). TEE can execute smart contract for keeping data private and represent off-chain data...
securely [15]. Many scholars have proposed a variety of approaches to combine TEE with blockchain.

Zhang et al. [16] proposed Opaque, a distributed data analysis platform that supports many queries and provides strong security guarantees. Opaque provided strong security guarantees including computation integrity and obliviousness by using the Intel Software Guard Xensions (SGX) hardware enclave. Hunt et al. [17] presented Ryoan, which protects sandbox instances from potentially malicious computing platforms by Intel SGX enclaves. The protected sandbox instances restricted untrusted data-processing modules to prevent leakage of users’ input data. Dang et al. [18] proposed optimizations that harness Intel SGX to improve upon the state-of-the-art solutions. The authors provided two design principles, namely scale up by scaling down and prioritize consensus messages that enable the consensus protocols to scale. Schuster et al. [19] proposed VC3, the first system that allows users to run distributed MapReduce computations in the cloud. VC3 adopted Intel SGX processors to isolate memory area on a single computer and deployed new protocols to secure distributed MapReduce computations. However, these systems cannot address state integrity and confidentiality over a long-lived system.

Bowman et al. [20] presented Private Data Object (PDO), a technology that enables mutually untrusted parties to run smart contracts over private data using Intel SGX. PDO used an interpreter enclave that executes a smart contract written in scheme. R3 Corda Distributed Ledger Platform [21] proposed Corda, a distributed ledger for recording and managing financial agreements. Some aspects of transaction validation were taken place in Intel SGX and may run on untrusted nodes. Microsoft [22] proposed Coco, a framework for consortium blockchain built on top of a network of hardware-protected TEEs. Coco integrated distributed ledger state, consensus algorithms, and a runtime environment for executing Ethereum smart contract in Intel SGX. Cheng et al. [23] proposed Ekiden, a system that combining blockchain with TEE to provide a generic platform for confidentiality-preserving contracts. Ekiden can support blockchain systems with non-final consensus by using a probabilistic Proof-of-Publication protocol which relies on trusted timers. However, these systems try to put the entire blockchain nodes and smart contracts computation in Intel SGX, which may introduce large attack surfaces, cause security issues and increase some runtime overhead.

Tran et al. [24] proposed Obscuro, an efficient and secure Bitcoin mixer that utilizes TEE. Obscuro can ensure the correct mixing operations and the protection of sensitive data with the TEE’s confidentiality and integrity guarantees for code and data. Taxa team [25] proposed Taxa Network, a decentralized computing network supported by the state-of-the-art trusted computing. The authors described a decentralized approach to the Taxa trusted computing network, such as the implementation of trusted computing nodes, and the PBFT-derived consensus mechanisms. However, these systems are not suitable for large, fast and complex computing enterprise blockchain systems.

Zhang et al. [26] proposed Town Crier, an authenticated data feed for smart contracts designed specifically to support Ethereum. Town Crier integrated the blockchain front-end with the trusted hardware back-end to scrape HTTPS enabled websites and serve source-authenticated data that relies on smart contracts. The use of Intel SGX allows Town Crier to serve datagrams with a high degree of trustworthiness. Brandenburger et al. [27] proposed an architecture and prototype for executing smart contracts within Intel SGX for Hyperledger Fabric. The authors explored issues that arise from the combination of TEEs with blockchains and presented a solution that handles rollbacks for Intel SGX. However, these systems did not consider the memory limits of SGX and the storage efficiency of blockchain.

Kosba et al. [28] proposed Hawk, a smart contract system that provides confidentiality through executing contracts off-chain and releasing zero-knowledge proofs on-chain. However, Hawk’s zero-knowledge proofs will generate high computational overhead. Hawk was designed for a single computed node and cannot provide high availability.

IBM Blockchain Platform [29] proposed an enterprise blockchain solution that allows the use of Secure Service Container (SSC) to deploy Fabric network. The platform ran the entire peer and its Linux operating system in a secure container, the container is not accessed by the host and host administrators. This technology required specialized mainframe hardware and has a large energy consumption.

Joshua Lind et al. [30] proposed Teechain, the first blockchain-based payment network that can operate with asynchronous blockchain access and offer dynamic collateral assignment. It leveraged Intel SGX to establish stateful payment channels among mistrusting parties. The off-chain channel can resolve transactions without having to conduct blockchain transactions for each transaction. Teechain attempts to solve the payment network problems of permissionless blockchain, whereas we try to address the privacy and performance issues of permissioned blockchain.

As a result, existing TEE-based blockchain schemes suffer from various deficiencies, such as high computational overhead, large attack surface, low availability and memory limitations of Intel SGX. They cannot meet the confidentiality and high performance requirements of DApps simultaneously. Therefore, it is essential to design an optimized blockchain system to support confidentiality along with high performance.

III. PRELIMINARIES

In this section, we briefly introduce blockchain and TEE with Intel SGX and explain some important features of them for which our proposal relies on. Experienced blockchain and Intel SGX readers can skip this section.
**A. BLOCKCHAIN**

Blockchain was proposed by Satoshi Nakamoto in 2008 for the use in cryptocurrency bitcoin [31]. Blockchain has a special structure consisting of blocks and chains that provides many unique advantages, such as transparency, verifiability, traceability and irrevocability [32]. Blocks hold batches of valid transactions that are hashed and encoded into a Merkle tree. Each block contains a hash of the previous block, a timestamp, a Merkle Root of transaction data and several transactions. This iterative process confirms the integrity of the previous block and goes back to original genesis block [33]. Once stored in the blockchain, the integrity and auditability of the data remain protected as long as the blockchain exists. If a previously published block is changed, it will have a different hash value. This mechanism makes it possible to easily detect and reject altered blocks. Figure 1 shows the blockchain data structure consisting of three blocks.

![Blockchain data structure consisting of three blocks.](image)

The two main functions of Intel SGX are sealing and remote attestation.

1) **SEALING**
Sealing is a process of storing sensitive information outside the enclave when the system shutdown [40]. SGX provides a sealing key to encrypt data and protect its integrity. Only after the trusted environment is restored, can the sealed data be unsealed. By calling EGETKEY instruction, the current enclave can access the sealing key, seal the data and export the data to a memory region outside the enclave [41].

2) **ATTESTATION**
Remote attestation is the security mechanism by which a remote party can verify whether a piece of code is running in an enclave of the Intel SGX platform [42]. Remote attestation in Intel SGX is a two-step process. In the first step, the enclave to be attested proves its identity to the system enclave present on every platform and called quoting enclave [14]. In the second step, the software that created the enclave can request a quote, which includes the report and its signature signed with a hardware-protected attestation key [43]. IAS mediates communication between quoting enclaves and remote verifiers, and it only allows the registered party to issue remote...
attestation requests. Remote parties can verify the quote by contacting the IAS.

IV. THE PROPOSED TEE-BASED BLOCKCHAIN SCHEME

A. SYSTEM ARCHITECTURE

To enhance the confidentiality and performance of permissioned blockchain, we present Hybridchain, a hybrid architecture combining TEE and blockchain. Figure 3 depicts the architecture of Hybridchain. As it shows, there are three types of entities: clients, computed nodes and consensus nodes.

![Figure 3. Architecture of Hybridchain.](image)

- Clients are end users of confidentiality-preserving smart contracts. In our scheme, clients can create contracts or execute existing ones. Clients delegate computation requests to nodes, which are executed within TEEs at any computed node.
- Computed nodes are off-chain nodes with special TEE-enabled CPUs, which can run smart contracts. Computed nodes are required to have high computation capability to meet the diverse and complex computation requirements of DApps. A computed node consists of REE and TEE. REE includes transaction/query interface and storage of transaction data. TEE includes transaction validator, smart contract execution in virtual machine (VM), and memory cache. The client’s computation proposal is first sent to the computed node, the computed node executes the transaction proposal and the corresponding smart contract inside the enclave. Subsequently the computed node sends the transaction execution result to all consensus nodes for the signature verification of the signed output result. All consensus nodes verify the output result and return the signed endorsement. Computed nodes also perform key management of smart contract through key manager. The key manager will create or retrieve existing keys upon request. Memory cache is used in enclave to reduce data transfer costs between trusted area and untrusted area.
- Consensus nodes are committers in the blockchain network, responsible for the validation and packaging confirmation of transactions. Contract state and attestations are persisted on this blockchain. Consensus nodes are used to check the validity of state updates submitted by computed nodes using TEE attestations. Consensus nodes are also responsible for maintaining a distributed ledger.

B. HIERARCHICAL NETWORK

1) HIERARCHICAL NETWORK STRUCTURE

For the permissioned blockchain, if all nodes need to repeat all the processes of the transaction in order to verify its legitimacy, we call this transaction an on-chain transaction. If some logic of a transaction is executed outside the blockchain, and after submitting that transaction to the blockchain, the blockchain nodes do not need to repeat the execution of this off-chain logic, then we call the transaction an off-chain transaction. For an off-chain transaction, the blockchain is usually responsible for bookkeeping, while complex logical transportation take place off-chain. Some dedicated nodes can execute the transaction and other nodes can endorse the transaction executed by the dedicated nodes.

By decoupling the execution of smart contracts from the on-chain consensus mechanism and only putting the execution results of smart contracts instead of all on the chain, we can avoid the computational burden and latency of on-chain execution. As a result, our blockchain has the ability to support computation and storage of complex smart contracts.

If we want to decouple the execution of smart contracts from the on-chain consensus mechanism and execute smart contracts off-chain, we will face three problems: (i) How to ensure blockchain nodes execute the codes we gave it? (ii) How to ensure that the results of blockchain nodes execution are correct and have not been tampered with? (iii) If we want to protect the privacy of transaction data, how can we ensure blockchain nodes have not secretly stored a copy of our private data?

The key to solving these problems is TEE. We choose Intel SGX as the TEE provider, which is a special hardware-level TEE. Intel SGX shields the operating system, hypervisor, and BIOS from the enclave memory, which can protect the confidentiality and integrity of smart contract computation. Intel SGX generates evidence that can prove the correct and integrity of the computation off-chain through remote attestation and digital signature.

We separate the computation from consensus layer and design a novel hierarchical network structure. As shown in Figure 4, the hierarchical network structure of Hybridchain consists of four layers: data layer, consensus layer, computation layer and application layer. The data layer seals the underlying data storage and encryption techniques of blockchain, including data blocks, chain structure, timestamp, hash functions, Merkle trees and digital signature. The consensus layer consists of transmission protocol and some consensus algorithms such as Practical Byzantine Fault Tolerance (PBFT). The consensus layer is also responsible for the verification of execution results. The computation layer is responsible for transaction verification, smart contract execution in the VM and key management of the system.
FIGURE 4. Hierarchical network structure of Hybridchain.

The application layer is a programmable implementation of blockchain, sealing some script code, smart contracts and algorithms.

2) OFF-CHAIN COMPUTATION
Client sends computation request to computed node that registered in the blockchain network, including cid of the contract, request signature (Sig), etc. Computed node verifies the signature of the transaction sender, pulls the contract code on the chain according to the cid of contract. When the transaction data and contract code are ready, they are initialized and executed in Intel SGX enclave. After the execution is completed, the new state, the execution result and the transaction attestation are packaged into a transaction and sent to the consensus network. The off-chain computation function is defined as follows.

\[ k = (\text{state}, \text{result}, \text{attestation}) = f(\lambda, \text{Dec}(\text{cid})) \]  

where \( \lambda \) is the transaction data; Dec is the decryption function; state includes pre-state and new state; result is the output result of transaction data and smart contract execution at computed node; attestation is the proof of correctness and integrity of transaction.

3) ON-CHAIN VERIFICATION
When consensus nodes receive the transaction sent by computed node, they verify the signature (Sig) of the request sender, the secure computation attestation (\( \varphi \)) and the hash value (\( H(\text{State}_{\text{prev}}) \)) of the current state. Once the verifications are passed, they join the transaction pool and broadcast the transaction throughout the consensus network. Consensus nodes package the transaction and form the consensus.

After executing the contract in the enclave of TEE, The attestation (\( \delta_{\text{TEE}} \)) is generated, which is then sent to IAS by the computed node to generate the \( \varphi \).

\[ \varphi = (\alpha, \delta_{\text{TEE}}, \delta_{\text{IAS}}) \]

where \( \alpha \) is the validity of the \( \delta_{\text{TEE}} \), \( \alpha \in \{0, 1\} \); \( \delta_{\text{IAS}} \) is the signature of the IAS for \( \alpha \) and \( \delta_{\text{TEE}} \).

In this way, the on-chain verification function is defined as follows.

\[ K(T) = K(\varphi) \land K(H(\text{State}_{\text{prev}})) \land K(\text{Sig}_\alpha) \]  

C. WORKFLOW OF HYBRIDCHAIN
The workflow of Hybridchain includes three phases: key generation and registration phase, computation phase and verification phase.

1) KEY GENERATION AND REGISTRATION PHASE
In this phase, the following activities will be performed:
- Key Generation: RSA key pair is generated at client, consensus nodes and SGX of computed node.
- Registration: Client cid and client public key (pk) are shared to the SGX. Similarly, Consensus nodes cid and consensus nodes pk are shared to the SGX; SGX pk is shared with client and consensus nodes for securing the subsequent communications.

Figure 5 depicts the flow of key generation and registration of client. Since the key generation and registration process of consensus nodes is the same as for the client, we will not list them separately.

2) COMPUTATION PHASE
In computation phase, we use session key (sek) and signature (Sig) to ensure the security of communication between client and Intel SGX enclave. The sek is randomly generated symmetric key. A new sek is generated for each transaction. Sig can help in avoiding Modification attack and Man-In-The-Middle (MITM) attack in case of transaction contract sharing process. After SGX enclave receives the transaction contract, it executes the contract and sends the result back to client. Figure 6 depicts the flow of secure computation process between client and computed node.
3) VERIFICATION PHASE
The main function of the verification phase is to verify the proof and commit the transaction on to the blockchain. Figure 7 depicts the flow of secure verification process between computed node and consensus nodes.

D. MEMORY EXTENSION FOR INTEL SGX
1) MOTIVATION
Intel SGX can provide secure execution even against an untrusted operating system and hardware-based attacks. Enclave code and data are placed in a special memory area called Enclave Page Cache (EPC). Keys are generated at boot-time and are stored within the CPU. Pages are only decrypted when inside the physical processor core. External reads on the memory bus can only observe encrypted data. The size of the Intel SGX EPC is limited to 128 MB in current release of the Intel SGX specification. Among those, only 93 MB is available for applications and the remaining 35M is reserved for enclave management. If the memory usage of enclave is larger than the EPC limit, some pages will be evicted from EPC and remapped to non-EPC memory region with page granularity encryption and integrity protection. Such demand paging can cause a significant performance loss.

To alleviate the memory limitation of Intel SGX, secure database designs using Intel SGX have been proposed to guarantee confidentiality and integrity. A common approach is to put the entire or partial database codes into SGX [44]. However, this approach incurs challenges due to the Intel SGX limitations (e.g., larger TCB and database functionality). Another approach is to put the whole key-value storage codes outside of SGX and allow Intel SGX application to load/store encrypted secret data from/to key-value storage [45]. While it ensures the confidentiality of data stored in the untrusted region, it also discloses the database codes to attackers. The compromised key-value storage may be attacked intentionally without being noticed.

However, existing methods are not suitable for maintaining a large blockchain ledger, so we propose a new secure key-value storage system. This system allows blockchain applications running in Intel SGX to securely store transaction records to the whole key-value storage codes placed outside of Intel SGX. Rather than relying on the page-oriented enclave memory extension provided by Intel SGX, our key-value storage solution provides application-specific data protection. And through fine-grained encryption and integrity protection for each key-value pair, the main data structure is maintained in the unprotected memory area.
2) SECURE KEY-VALUE STORAGE FOR SGX

We use a hashed index structure with chains. The primary hash table that stores the key and data pairs is placed in an unprotected memory area. When an access to a key occurs, the corresponding key-value pair is searched from the unprotected memory area. Since the primary data structure is not protected by Intel SGX, each data entry must be encrypted and written to the primary hash table for set operations.

To encrypt each key-value pair, we use AES-CTR counter mode encryption with a per-key counter incremented for each encryption cycle. Figure 8 shows the data structure for a single data entry that supports encryption and integrity verification. The entry consists of KEY Hint, Encrypted KEY/VALUE, MAC and IV/CTR. To hide the key and value of each entry, the key and value are concatenated and encrypted by the counter mode encryption with the 128-bit global secret key and a per-key IV and counter. Although the key and value in the data entries are encrypted and not revealed directly, it is possible that the attacker can extract the distributions of keys per hash-bucket by investigating chained data entries with linked lists. By using a keyed-hash function for the hash index, the information leakage can be minimized from the hashed key distribution.

For the AES counter-mode encryption and CMAC hash computation, sgx_rijndael128_cmac and sgx_aes_ctr_encrypt in the Intel SGX SDK are used. Since IV and counter are managed in a combined manner in the sgx_aes_ctr_encrypt function, we also store IV and counter as a combined field.

Once the data entry is updated, IV/counter stored in the data entry is incremented. When a new entry is created, the IV/counter value is set to a randomly generated value by sgx_read_rand and stored in the data entry in plaintext.

For the set operation, the corresponding data entry is searched in the same way as for the get operation. If the same key exists, the key value field is updated with the encrypted new key and value. The set operation also requires reading the MACs of the other key-value pairs in the same bucket set to update the hash value. During the update, the IV/counter value of the key is incremented. Otherwise, the new entry is created and inserted to the head of the bucket. All of these computations are done inside the enclave and the updated data entry is written to the untrusted area.

V. SECURITY ANALYSIS

The combination of TEE and blockchain enables Hybridchain to have a small attack surface. The CPU boundary becomes the periphery of the attack surface. All data, memory, and I/O outside the periphery are encrypted. Hybridchain not only can guarantee the confidentiality of transaction data, but also can resist various attacks.

A. CONFIDENTIALITY

In order to ensure the confidentiality of data, external data is encrypted with the public key of the contract before being sent to computed nodes. In enclave mode, operations such as key generation, key computation, and data decryption are performed to ensure the external data and key are invisible and inaccessible to the outside world, so as to ensure the confidentiality of data.

B. RESISTANCE TO DISTRIBUTED DENIAL OF SERVICE (DDoS) ATTACK

It is possible for some specific TEEs to suffer from DDoS attacks. Attackers can prevent communication between network endpoints, eclipse nodes to prevent blockchain access, and force failures in the committee chain to reduce performance. Hybridchain inherits the advantages of blockchain and defends against DDoS attack through asynchronous blockchain access and fault tolerance.

C. RESISTANCE TO REPLAY AND STATE FORKING ATTACK

Hybridchain mitigates against replay and TEE state forking attacks. On the one hand, Hybridchain establishes secure network channels using public/private keys generated inside the TEE to prevent messages from being shared between different TEE instances; On the other hand, Hybridchain employs a fault-tolerant mechanism that protects against Byzantine TEE using committee chains.

D. RESISTANCE TO MODIFICATION ATTACK

Suppose the attacker modifies the broadcasted transaction or replied data, the behavior will be discovered and refused because of signature and message authentication code.

E. RESISTANCE TO MITM ATTACK

MITM attack is a widespread means of modifying or disguising a message to achieve the purpose of the attack. The communication between computed nodes and storage outside...
of TEE are encrypted by the AES-CTR counter mode and its integrity is verified by comparing MAC. The middle man cannot forge or replace it.

F. RESISTANCE TO SIDE CHANNEL ATTACK

Although TEE can protect confidentiality, recent studies have found data leakage caused by side channel attack. Attackers come from non-enclave parts, including operating system, System Management Mode (SMM), hypervisor, Basic Input Output System (BIOS) and other privileged software. Since the enclave has a large amount of resources shared with external non-enclave, this provides a rich side attack surface for attackers. To remedy this problem, many researchers have proposed different schemes, such as [46], [47]. These methods mainly get data from attack surface, derive control flow and data flow information, and then deduce enclave-protected data. Most of the current side-attack attacks affect the performance of the enclave, so it is possible to prevent these attacks by detecting time anomalies.

VI. PERFORMANCE EVALUATION AND COMPARISON

We implemented a Hybridchain prototype using Hyperledger Fabric v2.0 and Intel SGX SDK v2.0 for Linux to implement the chaincode enclave and the ledger enclave in C/C++. Computed nodes were with 1.8GHz and four-core Intel i7-8565U CPU that provides Intel SGX support. Each node was equipped with 8GB of memory and 2Gbps network connection. All nodes run on Ubuntu 18.04 LTS 64bit operation system. The enclave transaction validator is written in Go. The ledger state and the transaction proposal and proposal response are encrypted by RSA. We use AES-CTR and HMAC-SHA256 with a verification key generated to protect the interaction between enclave and storage outside of TEE.

In order to demonstrate and evaluate the performance of Hybridchain, we have implemented a sealed-bid auction use case. We built sealed-bid auction use case on Intel SGX in simulation mode that runs Hybridchain’s contract codes inside the enclave. It signs the hash of the output using enclave signature key. The auction chaincode runs in the Intel SGX enclave, which can ensure the confidentiality of bidders and bid values along with verification of correct auction. Smart contract produces a cryptographic signature over all bidders and thereby certify the auction process. The bidders commit their encrypted bids to the blockchain and the enclave decrypts them for determining the winner. When creating a new auction, the chaincode accesses the key-value storage codes and ensures that no auction with the same name already exists.

The entire sealed-bid auction consists of four phases: Deposit, Bidding, Validation and Open phase. In Deposit Phase, users sign the registration agreement and pay the deposit according to the system prompts. In Bidding Phase, users generate a HMAC of the bid value and encrypt the bid using RSA and submit it to the computed node. During Validation Phase, the computed node decrypts the bids and validates the integrity of the bid value using the HMAC and finds the largest bid. Subsequently the computed node sends the encrypted largest bid to all consensus nodes for the signature verification of the signed output result. All consensus nodes verify the output result and return the signed endorsement to computed node. In Open phase, the computed node announces the largest bid and the winning bidder according to the signed endorsement. The Validation and Open phases are executed in the enclave and the result of the bid selection is signed using the enclave generated private key.

We evaluated the performance of the sealed-bid auction by running sealed-bid auction algorithms with different number of submitted bids. In our use case, there will be many clients bidding on a given challenge and encrypting it. Ref. [27] proposed an architecture for executing smart contracts within Intel SGX for Hyperledger Fabric. The authors address difficulties posed by Hyperledger Fabric’s specific execution-order-verification architecture, preventing rollback attacks on TEE-based execution. Since Ref. [27] have evaluated the response latency for the evaluate transaction for different numbers of submitted bids, we used the same experimental environment and parameters for comparison. The specific experimental results are shown in Figure 9.

The entire sealed-bid auction consists of four phases: Deposit, Bidding, Validation and Open phase. In Deposit Phase, users sign the registration agreement and pay the deposit according to the system prompts. In Bidding Phase, users generate a HMAC of the bid value and encrypt the bid using RSA and submit it to the computed node. During Validation Phase, the computed node decrypts the bids and validates the integrity of the bid value using the HMAC and finds the largest bid. Subsequently the computed node sends the encrypted largest bid to all consensus nodes for the signature verification of the signed output result. All consensus nodes verify the output result and return the signed endorsement to computed node. In Open phase, the computed node announces the largest bid and the winning bidder according to the signed endorsement. The Validation and Open phases are executed in the enclave and the result of the bid selection is signed using the enclave generated private key.

We observe that our solution has a lower latency than Ref. [27] with the same number of submitted bids. As the increase of bidding clients, the performance of our solution under different number settings remains stable, which shows that our platform has good scalability under the premise of confidentiality.

We also evaluated the end-to-end throughput and latency for the entire transaction flow including registration phase, computation phase, and verification phase. The specific experimental results are shown in Figure 10 and Figure 11.

We observe that execution in Intel SGX of computed node reaches 0.96x–0.99x of the throughput achieved by the native blockchain execution. The delay of Hybridchain is slightly higher than that of native blockchain. This is due to our blockchain system consist of three phases, only one block confirmation time are reduced. The computational overhead time in proof generation and validation is more than one block.
outside of TEE. Analysis and experiments show that Hybridchain can support confidentiality-preserving along with high performance.

Further studies are still needed in the future. We will design a distributed key management mechanism to further improve the overall security of Hybridchain. In addition, we plan to extend Hybridchain to support the distributed parallel processing of complex and real-time computation.

**REFERENCES**

[1] M. Pourvahab and G. Ekbatanifard, “An efficient forensics architecture in software-defined networking-IoT using blockchain technology,” *IEEE Access*, vol. 7, pp. 99573–99588, 2019.

[2] S. Kim, “Two-phase cooperative bargaining game approach for shard-based blockchain consensus scheme,” *IEEE Access*, vol. 7, pp. 127772–127780, 2019.

[3] A. Lei, H. Cuiickshank, Y. Cao, P. Asuquo, C. P. A. Ogah, and Z. Sun, “Blockchain-based dynamic key management for heterogeneous intelligent transportation systems,” *IEEE Internet Things J.*, vol. 4, no. 6, pp. 1832–1843, Dec. 2017.

[4] M. Kuzlu, M. Pampantasopoulos, L. Gurses, and S. Rahman, “Performance analysis of a hyperledger fabric blockchain framework: Throughput, latency and scalability,” in *Proc. IEEE Int. Conf. Blockchain* (Blockchain), Jul. 2019, pp. 536–540.

[5] E. Androulaki et al., “Hyperledger fabric: A distributed operating system for permissioned blockchains,” in *Proc. 13th EuroSys Conf.* New York, NY, USA: Association for Computing Machinery, 2018, pp. 1–15, Art. no. 30.

[6] H. Gao, Z. Ma, S. Luo, and Z. Wang, “BFR-MPC: A blockchain-based fair and robust multi-party computation scheme,” *IEEE Access*, vol. 7, pp. 110439–110450, 2019.

[7] S. Bowe, A. Chiesa, M. Green, I. Miers, P. Mishra, and H. Wu, “ZEXE: Enabling decentralized private computation,” in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2020, pp. 947–964.

[8] X. Li, Y. Mei, J. Gong, F. Xiang, and Z. Sun, “A blockchain privacy protection scheme based on ring signature,” *IEEE Access*, vol. 8, pp. 76765–76772, 2020.

[9] U. Lee and C. Park, “SoftTEE: Software-based trusted execution environment for user applications,” *IEEE Access*, vol. 8, pp. 121874–121888, 2020.

[10] M. Park, J. Kim, Y. Kim, E. Cho, S. Park, S. Sohn, M. Kang, and T. T. Kwon, “An SGX-based key management framework for data centric networking,” *IEEE Access*, vol. 8, pp. 45198–45210, 2020.

[11] X. Zhu and Y. Badr, “A survey on blockchain-based identity management systems for the Internet of Things,” in *Proc. IEEE Int. Conf. Internet Things (iThings) IEEE Green Comput. Commun. (GreenCom) IEEE Cyber, Phys. Social Comput. (CPSCom) IEEE Smart Data (SmartData)*, Jul. 2018, pp. 1568–1573.

[12] A. Ahmad, K. Kim, M. I. Sarfaraz, and B. Lee, “OBLIVIATE: A data oblivious filesystem for intel SGX,” in *Proc. Netw. Distrib. Syst. Secur. Symp.* 2018, pp. 1–15.

[13] P.-L. Aublin, F. Kelbert, D. O’keeffe, D. Mathukumaran, C. Priebe, J. Lind, R. Krahm, C. Fetzer, D. Eyers, and P. Pietzuch, “TaLoS: Secure and transparent TLS termination inside SGX enclaves,” Imperial College London, London, U.K., Tech. Rep., 2017.

[14] C.-C. Tsai, E. D. Porter, and M. Vij, “Graphene-SGX: A practical library OS for unmodified applications on SGX,” in *Proc. USENIX Conf. Usenix Annu. Tech. Conf. (USENIX ATC)*, 2017, pp. 645–658.

[15] B. C. Xing, M. Shanahan, and R. Leslie-Hurd, “Intel software guard extensions (Intel SGX) software support for dynamic memory allocation inside an enclave,” in *Proc. Hardw. Architectural Support Secur. Privacy HASP*, 2016, pp. 1–9.

[16] W. Zheng, A. Dave, J. G. Beekman, R. A. Popa, J. E. Gonzalez, and I. Stoica, “Opaque: An oblivious and encrypted distributed analytics platform,” in *Proc. USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, 2017, pp. 283–298.

[17] T. Hunt, Z. Zhu, Y. Xu, S. Peter, and E. Witchel, “Ryoan: A distributed sandbox for untrusted computation on secure data,” in *Proc. 12th USENIX Conf. Operating Syst. Design Implement. (OSDI)*, 2016, pp. 533–549.
[18] H. Dang, A. Dinh, E.-C. Chang, and B. C. Ooi, “Chain of trust: Can trusted hardware help scaling blockchains?,” 2018, arXiv:1804.03099. [Online]. Available: https://arxiv.org/abs/1804.03099.

[19] F. Schuster, M. Costa, C. Fournet, C. Gkantsidis, M. Peinado, G. Mainer-Ruiz, and M. Russinovich, “V3C: Trustworthy data analytics in the cloud foundation,” in Proc. IEEE FOCUS, 2018, pp. 677–690.

[20] M. Bowman, A. Miele, M. Steiner, and B. Vavala, “Private data objects: An overview,” 2018, arXiv:1807.05686. [Online]. Available: https://arxiv.org/abs/1807.05686.

[21] M. Hearn. (2016). Corda: A Distributed Ledger. Whitepaper. [Online]. Available: https://docs.corda.net/static/corda-technical-whitepaper.pdf

[22] Microsoft. (2017). The Coco Framework Whitepaper. [Online]. Available: https://github.com/Azure/coco-framework

[23] R. Cheng, F. Zhang, J. Kos, W. He, N. Hynes, N. Johnson, A. Juels, A. Miller, and D. Song, “Eikiden: A platform for confidentiality-preserving, trustworthy, and performant smart contracts,” in Proc. IEEE Eur. Symp. Secur. Privacy (EuroS&P), Jun. 2019, pp. 185–200.

[24] M. Tran, L. Luu, M. S. Kang, I. Bentov, and P. Saxena, “Obscure: A bitcoin mixer using trusted execution environments,” in Proc. 34th Annu. Comput. Secur. Appl. Conf., Dec. 2018, pp. 692–701.

[25] Team Taxa. (2018). Taxa Network: A Universal Logic Layer for Blockchain. Whitepaper. [Online]. Available: https://docs.corda.net/view/9tpemqk

[26] A. Kosba, A. Miller, E. Shi, and Z. Wen, “Hawk: The blockchain model of cryptography and privacy-preserving smart contracts,” in Proc. IEEE Symp. Secur. Privacy (SP), May 2016, pp. 839–858.

[27] IBM. (2018). IBM Blockchain Platform. [Online]. Available: https://console.bluemix.net/docs/services/blockchain/index.html

[28] Tescchain: A Secure Payment Network With Asynchronous Blockchain. Accessed: Jul. 18, 2017. [Online]. Available: https://arxiv.org/abs/1707.05454v4

[29] S. Nakamoto. (2009). Bitcoin: A Peer-to-Peer Electronic Cash System. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[30] A. Dorri, S. S. Kanhere, R. Jurdak, and P. Gauravaram, “Blockchain for IoT security and privacy: The case study of a smart home,” in Proc. IEEE Int. Conf. Pervasive Comput. Commun. Workshops (PerCom Workshops), Mar. 2017, pp. 618–623.

[31] A. Davenport and S. Shetty, “Air gapped wallet schemes and private key leakage in permissioned blockchain platforms,” in Proc. IEEE Int. Conf. Blockchain (Blockchain), Jul. 2019, pp. 541–545.

[32] P. Thakkar, S. Nathan, and B. Viswanathan, “Performance benchmarking and optimizing hyperledger fabric blockchain platform,” in Proc. IEEE 26th Int. Symp. Modeling, Anal., Simulation Comput. Telecommun. Syst. (MASCOTS), Sep. 2018, pp. 264–276.

[33] A. Ouaddah, A. A. Elkalam, and A. A. Ouahman, “FairAccess: A new blockchain-based access control framework for the Internet of Things,” Secur. Commun. Netw., vol. 9, no. 18, pp. 5943–5964, Dec. 2016.

[34] A. Ouaddah, A. A. Elkalam, and A. A. Ouahman, “Towards a novel privacy-preserving access control model based on blockchain technology in IoT,” in Proc. Eur. MENA Cooperation Adv. Inf. Commun. Technol. Cham, Switzerland: Springer, 2017, pp. 523–533.

[35] T. M. Fernandez-Cames and P. Fraga-Lamas, “A review on the application of blockchain to the next generation of cybersecurity industry 4.0 smart factories,” IEEE Access, vol. 7, pp. 45201–45218, 2019.

[36] W. S. Melo, A. Bessani, N. Neves, A. O. Santin, and L. F. R. C. Carro, “Using blockchains to implement distributed measuring systems,” IEEE Trans. Instrum. Meas., vol. 68, no. 5, pp. 1503–1514, May 2019.

[37] F. Alder, A. Kurnikov, A. Paverd, and N. Asokan, “Migrating SGX enclaves with persistent state,” in Proc. 48th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN), Jun. 2018, pp. 195–206.

[38] S. Vauhkonen, R. Pieper, P. Felber, M. Pasin, V. Shiavoni, and C. Fetzner, “SGX-aware container environments for heterogeneous clusters,” in Proc. IEEE 38th Int. Conf. Distrib. Comput. Syst. (ICDCS), Jul. 2018, pp. 730–741.

[39] J. Van Bulck, M. Minkin, O. Weisse, D. Genkin, B. Kasikci, F. Piessens, M. Silberstein, T. F. Weinsch, Y. Yarom, and R. Strackx, “Foresight: Extracting the keys to the intel SGX kingdom with transient out-of-order execution,” in Proc. USENIX Secur. Symp., 2018, pp. 991–1008.

[40] J. Park, S. Park, J. Oh, and J.-J. Won, “Toward live migration of SGX-enabled virtual machines,” in Proc. IEEE World Congr. Services (SERVICES), Jun. 2016, pp. 111–112.

[41] L. Coppolino, S. D’Antonio, G. Mazzo, and L. Romano, “A comparative analysis of emerging approaches for securing java software with intel SGX,” in Proc. Future Gen. Comput. Syst., vol. 97, pp. 620–633, Aug. 2019.

[42] J. Park, S. Park, B. B. Kang, and K. Kim, “EMotion: An SGX extension for migrating enclaves,” Comput. Secur., vol. 80, pp. 173–185, Jan. 2019.

[43] I. Sfyrakis and T. Gross, “UnGuard: Protectingunikernels using intel SGX,” in Proc. IEEE Int. Conf. Cloud Eng. (IC2E), Apr. 2018, pp. 99–105.

[44] O. Oleksenko, B. Trach, R. Krahn, A. Martin, C. Fetzner, and M. Silberstein, “Vaurus: Protecting SGX enclaves from practical side-channel attacks,” in Proc. USENIX Conf. Usenix Annu. Tech. Conf. (USENIX ATC), 2018, pp. 227–239.

[45] G. Chen, W. Wang, T. Chen, S. Chen, Y. Zhang, X. Wang, T.-H. Lai, and D. Lin, “Racing in hyperspace: Closing hyper-threading side channels on SGX with contrived data races,” in Proc. IEEE Symp. Secur. Privacy (SP), May 2018, pp. 178–194.

YONG WANG is currently pursuing the Ph.D. degree in cyber security with the School of Cyber Science and Engineering, Wuhan University, advised by Prof. June Li. His research interests include cyber-physical security, blockchain, and trusted execution environment.

JUNE LI received the B.S. and M.S. degrees in electrical engineering and computer engineering from the Wuhan University of Hydraulic and Electric Engineering, Wuhan, China, in 1986 and 1989, respectively, and the Ph.D. degree in computer engineering from Wuhan University, Wuhan, in 2004. She is currently a Professor with the Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University. Her research interests include network architecture, cyber security, blockchain, cyber-physical systems, and security of power industrial control systems.

SIYU ZHAO is currently pursuing the M.S. degree in cyber security with the School of Cyber Science and Engineering, Wuhan University, advised by Prof. June Li. His research interests include cyber-physical security and blockchain.

FAIJANG YU received the B.A. and Ph.D. degrees from Wuhan University, China. He is currently a Vice Professor with the School of Cyber Security and Engineering, Wuhan University. His research interests include trusted computing and system security.

* * *