ENFORCING SECURITY AND PRIVACY IN DISTRIBUTED LEDGERS BY INTEL SGX

Xueping Liang

Department of Engineering and Computer Science, Virginia State University
1 Hayden St, Petersburg, VA 23806, USA
xliang@vsu.edu

Sachin Shetty
Peter Foytik

Virginia Modeling, Analysis, and Simulation Center, Old Dominion University
1030 University Blvd, Suffolk, VA 23435, USA
{sshetty,pfoytik}@odu.edu

Deepak Tosh

Department of Computer Science
The University of Texas at El Paso
500 W University Ave, El Paso, TX 79968, USA
dktosh@utep.edu

ABSTRACT

Distributed Ledger Technology (DLT) utilizes an architecture that can host a large number of nodes without pre-established trust to provide decentralized services. The blockchain is the most widely used architecture of distributed ledger, where transactions across the whole network are visible to all participants in a chain to prevent tampering. However, transactions may contain sensitive information such as business contract. To secure the system and protect user privacy, we propose a multi-channel architecture that leverages Intel Software Guard Extensions (SGX). We illustrate how SGX capabilities help to defend against attacks on distributed ledgers, by way of SGX enforcement on the participating machines. We adopt the design and implementation of a two layer architecture for securing the blockchain mining process and enhancing the transaction privacy. The security analysis and performance evaluation show that the design and protocols are capable of protecting privacy, defending against adversarial attacks and scalable.

Keywords: Intel SGX, Distributed Ledger, Blockchain, Security.

1 INTRODUCTION

Distributed Ledger Technology offers a range of benefits to public and private services (Walport, 2016), with one of the most important features being the decentralized architecture for data processing and distribution. Applications based on distributed ledger in cloud computing platform is widely adopted by the commercial and military environment to support data storage, on-demand computing and dynamic provisioning. Every node in the ledger network hosts a copy of the overall state across the ledger. The blockchain is a major implementation of distributed ledger technology to record transactions and consensus algorithms to achieve agreement. Cloud computing environments are complex and heterogeneous with a diversity of software and hardware components which are provided by different vendors, possibly introducing risks of vulnerabilities and incompatibility. Blockchain applications in cloud computing environment inevitably suffer from these vulnerabilities. Besides, the distributed and decentralized nature brings a potential of privacy and confidentiality risks, which are the main concerns for the adoption of blockchain applications by industrial and military scenario. Previous work (Liang, Shetty, Tosh, et al., 2017) implements a blockchain based application by using tamper-proof and immutable blockchain receipt generated for each transaction. However, there are still vulnerabilities existing in most distributed ledger implementations, including the Bitcoin network in which different attacks are identified and analyzed (Tosh, Shetty, Liang, et al., 2017). Some research work (Lind, Eyal, Pietzuch, & Sirer, 2016) proposed using trusted execution environment to protect the payment channel in blockchain networks so that attackers cannot steal funds without the capability to manipulate the transactions. The previous work focused on the security of the network and communication protocols but the secure execution of the program is neglected.

This paper makes a step forward and presents a multi-channel architecture based on Intel SGX to serve as a feasible solution and help defend effectively against adversarial attacks which are possible in distributed ledger system, as analyzed in this paper. The main contribution is presented in a two-layer architecture, including the SGX enabled connection layer and the multi-channel transaction layer. By enabling Intel SGX features before blockchain peer connection, the trusted execution environment is provisioned per platform. Each execution environment is isolated in an enclave which is a tamper-resistant zone in the memory, and even the administrator of the local machine cannot break into the enclave. Trusted communication is then provisioned by remote attestation between peers, ensuring the authorized participation and code execution. With the help of these two steps, the vulnerabilities can be effectively countered. The multi-channel scheme provides a method for isolating communication channels. With SGX enabled, the trusted participation of each peer in the communication channel is assured. Data transferred via the SGX enabled channel is kept confidential between channel members. Even if the channel member is compromised, data leakage from the trusted execution environment will be prevented.

2 BACKGROUND

To better illustrate the proposed approach to secure blockchain architecture with the adoption of Intel SGX hardware, an overview of the existing attacks and vulnerabilities that remain as potential risks for the adoption of blockchain into the current systems is given below, followed by the SGX capabilities.

2.1 Blockchain Vulnerabilities and Attacks

Due to the P2P communication model and the trustless node involvement, most blockchain services and blockchain-based applications could be vulnerable to certain attacks that are difficult to detect or prevent. Six possible vulnerabilities existing in distributed ledgers are presented and analyzed below.

Block Withholding Attack (A1). Blockchain nodes usually join a pool to mine blocks with other nodes. Once a new block is mined, the rewards are shared among all the pool members. However, some malicious nodes intend to join the pool but never publish the block that has been mined, which decreases the overall rewards of the pool (Rosenfeld, 2011). Such attack is hard to detect because the mining process is controlled by the owner of the mining platform. Even when the platform finds a solution to a new block, other nodes are not aware of the fact. Analysis (Courtois & Bahack, 2014) shows that block withholding attack makes it possible for a rogue miner to gain profit without effort.

Block Discarding Attack (A2). Block discarding attack (Bahack, 2013) happens when a node controls the majority of network connections with other nodes. A mined block needs to be confirmed by most of the nodes will be confirmed and added to the blockchain. If the network is controlled by a single party that intends to discard a certain block mined by a normal node, the attack will discourage the nodes in the controlled network from confirming this block. In this way, the newly mined block by the normal node will be set invalid if a block from other nodes is confirmed earlier than itself.

Replay Attack (A3). Replay attack often happens in P2P networks where there are frequent message exchange and propagation. Attackers may try to repeat or delay an intercepted data transmission, thus preventing the honest party from communicating normally with the party where the message comes from

(Dua, Gautam, Sharma, & Arora, 2013). Replay attack in blockchain network is discussed in several scenarios, such as smart grid (Kim, Song, & Jun, 2016) and Internet of Things environment (Lee & Lee, 2017), as well as blockchain based big data authentication protocols (Abdullah, Hakansson, & Moradian, 2017). Replay attack could also be leveraged to launch block withholding attack when the malicious node eavesdrops the block confirmation message and delays the message from immediate propagation. For blockchain applications where there are quantities of distributed nodes, it is challenging to mitigate such attacks using the traditional timestamping services since most message exchange is transferred in a simultaneous session and a concurrent manner.

Man-in-the-Middle Attack (A4). In the Man-in-the-Middle Attack (Callegati, Cerroni, & Ramilli, 2009), attackers try to intercept the message between two honest parties, possibly alter the message and deliver the modified message to both sides so that he can impersonate either side. Both two honest parties could leak sensitive information since they believe they are interacting with the authenticated parties. It is possible that a malicious node joins the block mining process and steal the mining result, that is, the solution to a puzzle in a proof of work blockchain, from another node and then pass it to other participants for confirmation. Even though there are methods to detect such attacks, but the timing is critical. When the node realizes that the solution is stolen, it is too late since the rewards are already distributed.

Potential Privacy Risks (A5). One major concern of blockchain is the privacy issue due to the distributed nature of node connection. Each node participates in the network actively to broadcast messages and receive rewards. Transactions made by each node is traceable by other nodes. This is also concerned with the anonymity issue which is acknowledged in Bitcoin transactions since the transactions are permanently recorded in the public ledger while everyone can see the detailed transaction balance and public addresses. Some scenarios such as financial institutes and military communications require isolated transactions built on top of tamper-proof blockchains. If the privacy issue is solved, users can communicate in a secure way without exposure of sensitive information and still make use of blockchain benefits for integrity protection.

Majority Hash Rate Attack (A6). To gain a large quantity of profit, some miners use special mining devices to control a great potion of mining power. When an attacker controls more than 50% of the entire mining power, he can reverse transactions he sent and even prevent some transactions from being confirmed, with a high possibility. Therefore, majority hash rate attack is also called 51% attack. This attack could also break the decentralized nature of blockchain since the majority is controlled by a powerful but single entity. A two-phase proof of work (Bastiaan, 2015) is proposed to prevent such attack but the hardware security is not seriously considered.

2.2 Intel SGX Security Capabilities

Intel SGX provides enclaves, which is an isolated zone for trusted program running, to reduce the attack surface and minimize the trusted computing base. We illustrate three key notions adopted by SGX, including Enclave, Attestation and Sealing, as follows.

Enclave. When an enclave is created, the sensitive code and data will be stored inside a protected memory region called Enclave Page Cache (EPC). The EPC region is encrypted which ensures strong confidentiality. The code inside the enclave is not authorized to access memory beyond the enclave boundaries. If code inside one enclave accesses content in another enclave, there would be an error of aborted page access. After the enclave is provisioned with appropriate memory content, there will be an enclave measurement stored in two registers, MRENCLAVE and MRSIGNER. An enclave author can issue several enclave certificates using the same RSA key to indicate different modules of the same software.

Attestation. Attestation is a process that one entity proves that it is running on top of a trusted platform to another. Intel SGX provides two types of attestation, including local attestation (two enclaves running on the same platform) and remote attestation (extending local attestation to outside of the platform). Local attestation can be performed between two enclaves: target enclave and challenger enclave. First, challenger enclave sends attestation request to target enclave for verification as well as challenger enclave's

MRENCLAVE value. Then target enclave uses EREPORT instruction as well as the received MRENCLAVE value to create a signed REPORT for challenger enclave and sends it back. Challenger enclave receives the REPORT and extracts the REPORT key to compute the MAC. If the MAC matches the value on the target enclave's REPORT, then challenger enclave can confirm that target enclave is running on the same trusted platform. Then challenger enclave can create a REPORT for target enclave and sends it to target enclave so that target enclave can confirm that challenger enclave is on the same platform in a similar way. In some cases, two enclaves on different platforms need to verify each other. Intel SGX enables a third enclave called quoting enclave to help launch a remote attestation. Same as local attestation, there is a challenger enclave requesting to verify the target enclave.

Sealing. When enclave program finishes running, the code and data inside the enclave will be gone. To store the data for future use, Intel SGX provides a sealing key to encrypt data and ensure data integrity. The sealed data can only be unsealed when the trusted environment is restored locally (Anati, Gueron, Johnson, & Scarlata, 2013). By calling EGETKEY instruction, current enclave can access the sealing key, seal the data and export the data to a memory region outside the enclave. There are two sealing policies designed by Intel, including sealing to the enclave identity and sealing to the sealing identity. Depending on the access control policies of the enclave applications, different sealing policies can be adopted. Sealing to the enclave identity can produce a key only available to the exact enclave instance. If a key available to different enclave instances under the same sealing identity is needed, the policy of sealing to the sealing identity (Sealing Authority) can make it. Based on the key characteristics above, the security capabilities of SGX can be summarized as in Table 1.

Table 1: Security Capabilities of SGX.

Intel SGX Capabilities	**Explanation**
Enclave Execution (C1)	Tamper-resistance against software attacks outside the enclave (McKeen et al., 2013)
Hardware-rooted Randomness (C2)	A valuable source for cryptographic key generation and protection. A true random function is provided in the tRTS (Trusted Run-Time System) library available in Intel CPUs (Aumasson, 2016).
Remote Attestation (C3)	Allows a client platform to attest itself to a remote party proving that the client is running in a trusted environment. This ensures the integrity of code execution in both the client and server side.
Trusted Elapsed time (C4)	Provides a hardware-assisted measure for trusted timestamping service, which is critical for scenarios where it is time sensitive.
Confidentiality Assurance (C5)	Prevents sensitive transactions and business contracts from leakage. The enclave identity key and provisioning key can be involved for secret protection and attestation.
Sealing and Unsealing (C6)	Helps to store confidential information outside the enclave for future access after system shutdown.
Monotonic Counter (C7)	Serves as a measure to defend against the replay attack. Altogether, these Intel SGX capabilities will greatly reduce the attack surface and minimize the trusted computing base.

3 DESIGN OF SGX-ENABLED DISTRIBUTED LEDGER WITH MULTI-CHANNEL ARCHITECTURE

Considering SGX security capabilities and the promising usage of enclave isolation, we use the SGX capabilities (C1-C7) mentioned above to defend against the six attacks (A1-A6) which are possibly existing in blockchain applications, following the basic design rationale as illustrated in Table 2.

Table 2: Design Rationale of Attack Oriented and SGX Enabled Blockchain.

Potential Attacks	**Intel SGX Capabilities**
Block Withholding Attack (A1)	Enclave Execution (C1)
Block Discarding Attack (A2)	Enclave Execution (C1), Remote Attestation(C3), Monotonic Counter(C7)
Replay Attack (A3)	Hardware-rooted Randomness (C2), Remote Attestation(C3), Trusted Elapsed time (C4)
MITM Attack (A4)	Hardware-rooted Randomness (C2), Remote Attestation(C3), Confidentiality Assurance(C5)
Potential Privacy Risks (A5)	Enclave Execution (C1), Confidentiality Assurance(C5), Sealing and Unsealing(C6)
Majority Hash Rate Attack (A6)	Enclave Execution (C1)

Trusted execution could be established in an enclave zone (C1) which provides an isolated environment and thus the reduced attack surface. The code and data inside the enclave are integrity-protected. Even when the platform and the Operating System is controlled by a malicious entity, the mining is still protected from compromise. This guarantees that once a block is mined, the block will be immediately submitted to the network, making block withholding attack difficult. Hardware-rooted Randomness (C2) helps to generate random numbers used in key management protocols, benefiting the message exchange in transactions between blockchain nodes and the resistance of replay attack. Remote Attestation (C3) is utilized during key negotiation and secret exchanges. The shared secrets established is platform dependent, making it effective for resisting the man-in-the-middle attack. To provide a message with a trusted timestamping, the SGX capability of trusted elapsed time (C4) calls *sgx_get_trusted_time()*. This trusted time can effectively prevent replay attack where a given expire time is set. Confidentiality Assurance (C5) protects peer to peer communication in the transaction process and preserve the confidentiality of both identity and transaction. Enclaves can seal and unseal the secret (C6) shared by two parties for multiple usages. Monotonic Counter(C7) provides a trusted counter by calling *sgx_create_monotonic_counter()* function, which is utilized to preserve message authenticity during communications, further enhancing resistance against replay attack and man-in-the-middle attack.

Based on the above design rationale, a further contribution is made to protect privacy by adopting a two-layer architecture, comprising an upper transaction layer featured with multiple channels and a connection layer enabled with SGX hardware, as is shown in Figure 1. The transaction layer uses multiple channels to separate different group of nodes. Each channel node is responsible for maintaining states of the current channel. Access to each channel is predefined during node enrollment. The connection layer will be used for the communication between nodes. The enclave process hosts and isolates the block mining code while the non-enclave process is responsible for broadcasting information among blockchain nodes.

3.1 SGX Enabled Connection Layer

In Figure 1, we design two processes for the program running of block mining and consensus in each node, enclave process and non-enclave process, to isolate critical code and data. We assume the enclave process is trusted and tamper-resistant against adversarial compromise. The communication between each node is established by the non-enclave process, which suffers from the above mentioned six attacks (A1-A6). There are four steps for the enclave provisioning and information exchange, as illustrated below while the authentication between the node and the network manager is out of the scope of this paper.

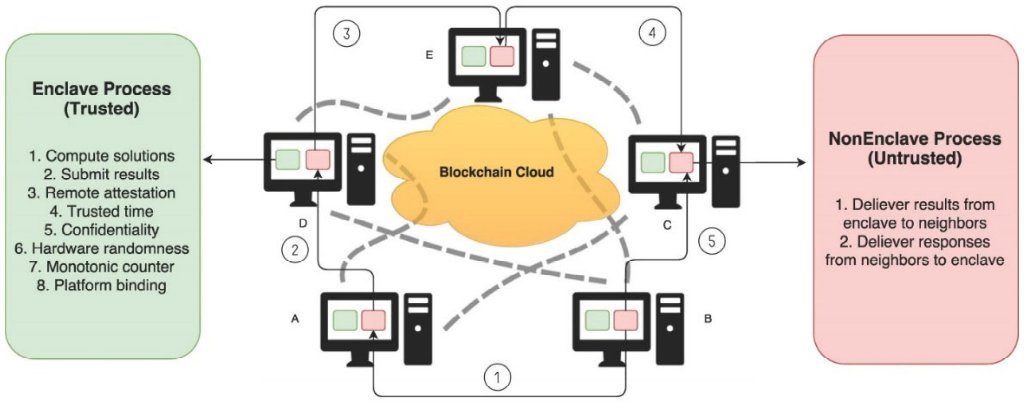

Figure 1: SGX Enabled Blockchain Architecture in the Cloud.

Node Involvement. To participate in the blockchain network, the node needs to connect to other neighboring nodes (or peers). During the first time when two nodes connect to each other, the remote attestation is launched with each node enabled by SGX. For example, in Figure 1, if peer A and B are trying to build a secure and trusted connection, they will request an attestation from each other. The attestation is launched in the following steps. First, peer A sends a request to B for attestation and the enclave running on peer B will generate a data structure called REPORT which is evidence for trusted enclave execution. The REPORT will be forwarded to a quoting enclave which is specific to the platform providing a signed QUOTE to be verified at the challenger platform (peer A). Peer A will request the Intel Attestation Service for QUOTE verification. If verification passed, the platform authenticity of peer B is confirmed. Likewise, peer A can be attested upon the request of peer B and other nodes perform similar actions as well for the first connection request. By remote attestation, the connection between two trusted platforms is established. Replay attack (A3) and Man-in-the-Middle attack (A4) can be mitigated since the message exchanged can be generated with a binding to the specific platform. If a middle man intercepts the message, the message origin and message sender will not be consistent. Any message replayed from another platform will be detected immediately. This phase is also responsible for key negotiation which will be used in the following message propagation including mined block broadcast from both neighbors and the peer itself. Due to performance considerations, the remote attestation is performed only when the peer receives the first connection request with the same peer. The following connection will only need to use the negotiated key for future communications since the key is originated from the authenticated party which is platform dependent, and is only known by the two attested peers. Only when the keys expire, then a new remote attestation process is needed.

Block Mining Isolation. To preserve the integrity of the mining code running and the mined solution, we keep the mining code inside the enclave (C1). This CPU-dependent mining also prevents the occupation of large numbers of resources which are the prerequisites of block withholding attack (A1) and majority hash rate attack (A6). By isolation, the enclave process will not be manipulated by the non-enclave process, which preserves the code integrity even when the non-enclave process is compromised. In this way, both block withholding attack and block discarding attack (A2) will be prevented since we can place the following two steps inside the enclave, including (1) mine the solution and submit the mining result regardless of whether there is a solution; (2) receive messages from neighbors regardless of whether it is a block mining solution or a transaction broadcast. All the process logic for the above two steps are handled inside the enclave, in case that the non-enclave process takes control and misbehaves. Note that it is important to include a nonce generated from the hardware (C2) in case of chosen plain-text attack, preventing the non-enclave process from guessing the message type.

A certain amount of blockchain transactions are grouped into one block. Each block contains the hash of the previous block and links to the previous block, making it a distributed and robust ledger in that multiple copies of the ledger is maintained among the participating miners. Integrity is preserved since it is technically hard to change a certain block with the necessity to change all the following blocks. Mining is the process to keep record of each block. Taking proof of work as an example consensus algorithm used in our design, the following algorithm (Pow) is used to isolate the mining process inside an enclave. Specifically, the enclave environment should be provisioned first by initialing the parameters required and then start to compute solutions for new blocks. If a valid block is mined successfully, the miner will notify the neighbors to broadcast the solution, which is encrypted. Once other peers receive a message from their neighbors, an enclave environment will be provisioned the same way to decrypt the message. If the block is valid, it will then notify its own neighbors for further propagation.

Mined Solution Submission. Once a peer finds the solution for a new block, it will submit to the pool manager for block confirmation and get rewarded. Commonly, miners join a pool and mine continuously to gain competitive computing power. There is a pool manager to assign mining tasks to pool members. Once the pool manager receives a new block solution from honest miners, the manager will broadcast the block to the blockchain network. If the blockchain network miners reach a consensus on the newly mined block, a certain amount of reward will be received by the pool manager. Hence, it is critical that the new block is recorded and the miner who finds the block is recorded, both in a trusted and tamper resistant manner. With the adoption of Intel SGX features, two security guarantees can be achieved. For one thing, once a mined block is passed on to a malicious peer, the malicious peer cannot ignore the block since he has no control over the program running inside an enclave (C1). For another, the block confirmation message is encrypted by the key negotiated in node involvement phase, so it is impossible to distinguish a block confirmation message from other messages since only the enclave process can decrypt the message using the same key negotiated during the remote attestation process. The block solution is also sealed for secure local storage (C6). This is effective for defending against block withholding attack, in that the malicious peer is oblivious about whether the enclave process has mined a new block or not, thus it is difficult to decide whether to withhold the message or not. Besides, the attempt to modify the message is impossible since once it is modified, the integrity is disrupted, and any valid message cannot be recovered. To defend against block discarding attack, the data propagation process is designed to run inside an enclave to avoid malicious data discarding. The block mining process runs inside the enclave to provide integrity of the process while the block submission is bound to the platform to achieve tamper resistance.

Mining Rewards Distribution. When a new block is confirmed, the peer who finds the new block will get rewarded. There are several reward schemes existing for pool based mining (Rosenfeld, 2011). In proportional reward scheme, the reward is distributed proportionally according to the shares submitted by each miner. The share is the work load needed to find the final solution, that is the interval between two valid blocks. In this case, some malicious miners may frequently hop from different pools during a competition round for maximizing the rewards. For peer platforms SGX enabled, the mining record can be tracked on a platform basis because both shares and solutions are generated from specific platforms. In this way, non-frameability is assured for the honest miners, and the greedy miners cannot be repudiated. Meanwhile, each share and solution submission are bound with a trusted elapsed time, which maximizes the trustworthiness of the node, providing a matrix for a fair reward distribution and an authenticated identity for confirming the deserved reward receiver. By using the platform dependent key for confirmation, all the neighbors of the peer that proposes the new block witnesses the reward distribution.

3.2 Multi-channel Transaction Layer

To preserve the privacy of participating nodes in the blockchain network, a multi-channel architecture is adopted to isolate transactions. Enclave isolation preserves the integrity of blockchain mining code and data, while channel isolation preserves the privacy of transactions by limiting the access to restricted participating nodes. Each channel maintains a ledger called subledger for efficient transaction signing and verification. Meanwhile, there is a system ledger maintained by all channels. For some blockchain implementations, such as Hyperledger Fabric (Androulaki, 2017) and R3 Corda (Brown, 2017), which support multiple channels, a further step is taken to combine the multi-channel scheme with Intel SGX to provide a privacy enhanced architecture. A specially assigned peer is responsible for generating the genesis block (the first block for the ledger) for each channel during channel establishment. Access control policy is defined in the genesis block configuration. To enhance user privacy, each channel hosts a certain number of peers while each peer joins the channel by binding the platform key with the channel ID which is publicly known to all nodes in the blockchain. SGX enabled channel can help defend attackers from hijacking a channel member by isolating channel transactions inside enclaves. The channel metadata and the platform key are used to bind a node to a channel so even an attacker that has access to the channel member's certificate cannot participate in the channel since there is no binding between the channel and the attacker platform. Meanwhile, Man in the Cloud (MITC) attack (Liang, Shetty, Zhang, Kamhoua, & Kwiat, 2017) is prevented by sealing confidential information locally.

In Figure 2, there is a system ledger and several subledgers, as well as three nodes. Each subledger represents one channel, such as Subledger 1 for Channel 1. Node A belongs to both Channel 1 and Channel 4. Both Node A and Node C participate in Channel 4. Therefore, blocks in Channel 4 are signed by both nodes. However, due to channel isolation, Node A is oblivious to transactions in Channel 2. The binding between the node and the channel is also included in a transaction which is public to all channel nodes, and stored in the node platform using SGX sealing feature (C6). The attacker who even holds the certificate maliciously cannot be involved in attempted channel mining because it will be easily detected that the certificate is originally from another platform instead of the attacker. This preserves both privacy of channel members and the integrity of the permission to participate in the channel.

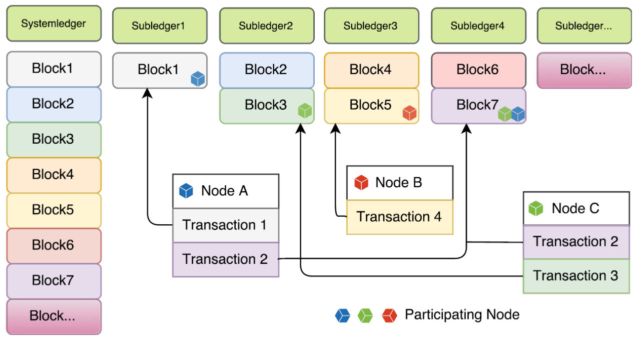

Figure 2: System Ledger and Subledger.

In Hyperledger Fabric (Androulaki, 2017), there are enrollment certificate (ECert) and transaction certificate (TCert). Both certificates are signed by the Certificate Authority (CA) which is part of the membership service provider. On receiving certificate issuance request from potential channel members, the CA will build a trusted connection with the node using SGX remote attestation and obtain the platform-dependent key, noted as $\boldsymbol{K_{platID}}$. To simplify the binding of a specific node to the channel and make certificates platform dependent, $\boldsymbol{ECert_{SGX}}$ is used to represent the platform self-signed certificate (or token) for new member enrollment.

$$\boldsymbol{ECert_{SGX}} = (\boldsymbol{ECert, platformID})_{\boldsymbol{sig}}$$

The $\boldsymbol{TCert_{SGX}}$ is used for transaction participation in the channel-isolated communication, which contains the identifier of the channel that the member has access to.

$$\boldsymbol{TCert_{SGX} = (TCert, platformID, channelIDs)_{sig}} .$$

The above certificate ensures that each participant is assigned an original certificate and generate a self-signed certificate accordingly. By doing so, the participating nodes can maintain the authorization from the CAs and providing additional security guarantee of the trusted platform. Most importantly, trusted members constitute a trusted channel with assured privacy preservation.

4 EVALUATION

This section provides an evaluation of the architecture in three aspects, namely the multi-channel performance, the hardware cost of enabling SGX and the time overhead of running program inside Intel SGX.

4.1 Multi-channel Transaction Performance and Hardware Cost

By channel isolation, the privacy of authorized transactors is preserved. By using different channel to transact with other nodes, the original transaction certificate is used for limited transactions to provide unlinkability. Therefore, the SGX secured certificates also provides untraceability for different transaction parties. The channel scalability is important in cloud applications where hardware requirement is concerned as a key factor. With the growth of node number, the number of channels scales exponentially.

Based on the effectiveness of the proposed architecture to defend against all six attacks and vulnerabilities, we move forward to evaluate the efficiency of the SGX enabled blockchain service performance. Intel SGX provides an isolated environment, protecting critical and vulnerable programs from compromise and malicious control. By enclave isolation, we preserve the integrity of code and data, as well as node privacy. With the growth of node number, the hardware cost grows linearly, while the number of channels scales exponentially. This shows the effectiveness for applications with a larger quantity of nodes, especially cloud applications built on top of distributed ledgers.

4.2 Time Overhead of Running inside Intel SGX

The concept of difficulty in proof of work represents the degree of the difficulty to which the mining task is set. The higher the difficulty value is, the more time is needed to compute and find a solution. To evaluate the time overhead with enclave execution, we utilize Intel SDK to test on real SGX machines and compute the time cost for the execution of proof of work algorithms inside an enclave. The test platform has an Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz (Skylake with 8MB cache) with 32GB RAM. The mining process uses different value of difficulty defined in the proof of work consensus algorithm. Considering the limited computing capability of a desktop machine, we suppose the value of difficulty ranges between 0 and 13, and observe the performance to gain a basic view of the performance. The time cost for SGX enclave execution is shown in Figure 3.

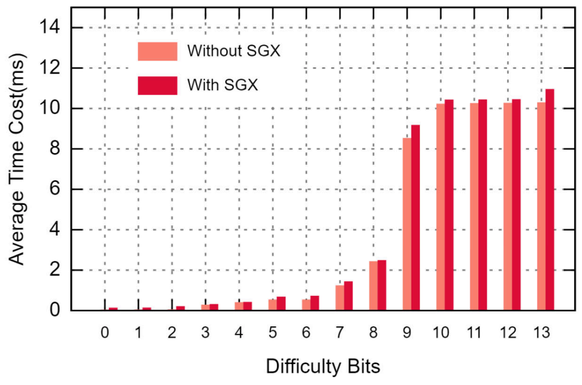

Figure 3: Average Time Cost with Different Difficulty Bits.

From Figure 3 we can conclude that SGX enabled mining algorithm brings an acceptable overhead to find a solution under a certain difficulty value, with an average overhead of 21%. The solution divided by the total time used to find the solution represents the hashing power. We also evaluate the average hashing power of SGX enabled machine against a non SGX enabled machine. Figure 4 shows that SGX enabled machine brings trivial effect on the hashing power.

The hashing power of SGX enabled machine decreases approximately 21.3% from non SGX enabled machine. This adds to the difficulty of the succeeding of a malicious node to launch various attacks and compromise the blockchain service. Therefore, if 51% of nodes are to be controlled by malicious parties, the hardware cost is considerably high and the time to modify a transaction will increase dramatically. This ensures that the security of SGX blockchain mining is enhanced with acceptable overhead, besides the preserved privacy and trust guarantee. The experiments are based on the Bitcoin network but could be extended to other implementations of distributed ledger. The exact overhead could be evaluated in real applications scenarios to measure the overhead brought by Intel SGX execution and the isolation process.

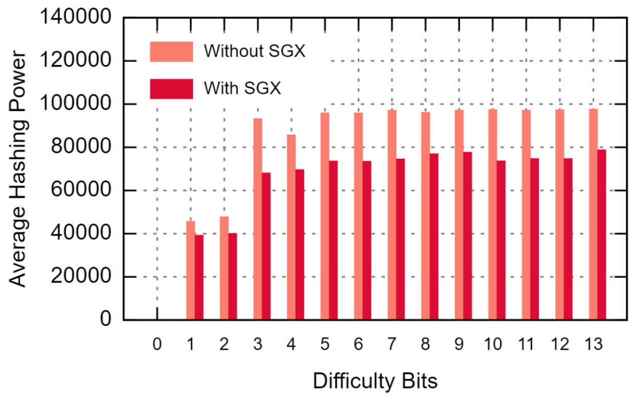

Figure 4: Average Hashing Power with Different Difficulty Bits.

5 RELATED WORK

Intel SGX offers a list of benefits from the cloud and cyber applications, which ensures a secure and trusted execution environment and reduces the attack surface. Trusted software solutions such as credential and password protection (Liang, Shetty, Zhang, et al., 2017) are implemented using Intel SGX to seal local privacy data and establish a trusted communication with cloud servers via remote attestation. Several SGX based blockchain architecture are proposed. Proof of Luck (Milutinovic, He, Wu, & Kanwal, 2016) provides an efficient consensus algorithm based on SGX as the trusted computing base. OpenSGX (Jain et al., 2016) is a software emulation platform based on QEMU but lacks security measures and capabilities. However, none of these work addresses the potential privacy risks.

Hawk (Kosba, Miller, Shi, Wen, & Papamanthou, 2016) is proposed to protect transaction privacy for smart contracts with the notion of cryptography primitives and a formal analysis model. However, the entire framework assumes of the input dependent privacy and the security of blockchain mining scheme. SGP (Tramer et al., 2017) is a cryptography primitive which can be applied in SGX based smart contract for transparent information exchange with fairness. Besides, research work (Kaptchuk, Miers, & Green, 2017) also emphasizes the adoption of SGX based blockchain and smart contract applications for security and privacy considerations, indicating the significant potential of the wide adoption of SGX enabled platforms.

6 CONCLUSION

This paper analyzes the potential vulnerabilities in blockchain architecture and utilize Intel SGX to help prevent these attacks. Meanwhile, a multi-channel scheme is adopted to effectively address the potential privacy issues, removing the need of trust between participating nodes during transactions. The evaluation shows that our architecture serves as a feasible solution with acceptable overhead. Future work will utilize the design, implement a privacy preserving blockchain service in a cloud data sharing application, and evaluate the overall application performance.

REFERENCES

Abdullah, N., Hakansson, A., & Moradian, E. (2017). Blockchain based approach to enhance big data authentication in distributed environment. In Ubiquitous and future networks (icufn), 2017 ninth international conference on (pp. 887–892).

Anati, I., Gueron, S., Johnson, S., & Scarlata, V. (2013). Innovative technology for CPU based attestation and sealing. In Proceedings of the 2nd international workshop on hardware and architectural support for security and privacy (Vol. 13).

Androulaki, E. (2017). Hyperledger. (https://www.hyperledger.org/)

Aumasson, L. (2016). Sgx secure enclaves in practice: Security and crypto review—kudelski security. Black Hat USA.

Bahack, L. (2013). Theoretical bitcoin attacks with less than half of the computational power (draft). arXiv preprint arXiv:1312.7013.

Bastiaan, M. (2015). Preventing the 51%-attack: a stochastic analysis of two phase proof of work in bitcoin. In Availab le at http://referaat. cs.utwente.nl/conference/22/paper/7473/preventingthe-51-attack-a-stochastic-analysis-of-two-phase-proof-of-work-in-bitcoin. pdf.

Brown, R. G. (2017). Introducing r3 corda: A distributed ledger designed for finanial services, 2016.

Callegati, F., Cerroni, W., & Ramilli, M. (2009, Jan). Man-in-the-middle attack to the https protocol. IEEE Security Privacy, 7(1), 78-81.

Courtois, N. T., & Bahack, L. (2014). On subversive miner strategies and block withholding attack in bitcoin digital currency. arXiv preprint arXiv:1402.1718.

Tosh, D. K., Shetty, S., Liang, X., Kamhoua, C. A., Kwiat, K. A., & Njilla, L. (2017, May). Security implications of blockchain cloud with analysis of block withholding attack. In Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (pp. 458-467). IEEE Press.

Dua, G., Gautam, N., Sharma, D., & Arora, A. (2013). Replay attack prevention in kerberos authentication protocol using triple password. CoRR, abs/1304.3550. Retrieved from http://arxiv.org/abs/1304.3550

Jain, P., Desai, S., Kim, S., Shih, M.-W., Lee, J., Choi, C., . . . Han, D. (2016). OpenSGX: An open platform for SGX research. In Proceedings of the network and distributed system security symposium, san diego, ca.

Kaptchuk, G., Miers, I., & Green, M. (2017). Managing secrets with consensus networks: Fairness, ransomware and access control. IACR Cryptology ePrint Archive, 2017, 201.

Kim, M., Song, S., & Jun, M.-S. (2016). A study of block chain-based peer-to-peer energy loan service in smart grid environments. Advanced Science Letters, 22(9), 2543–2546.

Kosba, A., Miller, A., Shi, E., Wen, Z., & Papamanthou, C. (2016). Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In Security and privacy (sp), 2016 ieee symposium on (pp. 839–858).

Lee, B., & Lee, J.-H. (2017). Blockchain-based secure firmware update for embedded devices in an internet of things environment. The Journal of Supercomputing, 73(3), 1152–1167.

Liang, X., Shetty, S., Tosh, D., Kamhoua, C., Kwiat, K., & Njilla, L. (2017). Provchain: A blockchain-based data provenance architecture in cloud environment with enhanced privacy and availability. In Proceedings of the 17th ieee/acm international symposium on cluster, cloud and grid computing (pp. 468–477).

Liang, X., Shetty, S., Zhang, L., Kamhoua, C., & Kwiat, K. (2017). Man in the cloud (mitc) defender: Sgx-based user credential protection for synchronization applications in cloud computing platform. In The 10th ieee international conference on cloud computing (cloud 2017).

McKeen, F., Alexandrovich, I., Berenzon, A., Rozas, C. V., Shafi, H., Shanbhogue, V., & Savagaonkar, U. R. (2013). Innovative instructions and software model for isolated execution. In Hasp@isca (p. 10).

Milutinovic, M., He, W., Wu, H., & Kanwal, M. (2016). Proof of luck: An efficient blockchain consensus protocol. In Proceedings of the 1st workshop on system software for trusted execution (p. 2).

Rosenfeld, M. (2011). Analysis of bitcoin pooled mining reward systems. arXiv preprint arXiv:1112.4980.

Sawtooth lake latest documentation. (2016). Retrieved from https://intelledger.github.io/introduction.html

Tramer, F., Zhang, F., Lin, H., Hubaux, J.-P., Juels, A., & Shi, E. (2017). Sealed-glass proofs: Using transparent enclaves to prove and sell knowledge. In Security and privacy (euros&p), 2017 ieee european symposium on (pp. 19–34).

Walport, M. (2016). Distributed ledger technology: Beyond blockchain. UK Government Office for Science.

AUTHOR BIOGRAPHIES

XUEPING LIANG is an Assistant Professor in the Department of Engineering and Computer Science at Virginia State University. She holds a PhD in Cyber Security from University of Chinese Academy of Sciences. Her research interests lie in cyber security and trusted computing, especially in blockchain and distributed computing. Her email address is xliang@vsu.edu.

SACHIN SHETTY is an Associate Professor in Old Dominion University. His research interests include cyber security, machine learning and the cyber physical security. His email address is sshetty@odu.edu.

PETER FOYTIK is a Senior Project Scientist at Old Dominion University. His research interests include modeling and simulation. His email address is pfoytik@odu.edu.

DEEPAK TOSH is an Assistant Professor in the Department of Computer Science at the University of Texas at El Paso. His research interests include cyber security and applied game theory. His email address is dktosh@utep.edu.