A Hybrid Blockchain Architecture for Privacy-Enabled and Accountable Auctions

Harsh Desai*, Murat Kantarcioglu*, Lalana Kagal†
*School of Computer Science
The University of Texas at Dallas
{hbd140030, muratk}@utdallas.com

†Computer Science and Artificial Intelligence Lab
Massachusetts Institute of Technology
lkagal@csail.mit.edu

Abstract— **Blockchain has recently emerged as an important tool that can enable critical distributed applications without requiring centralized trust. For example, public blockchains have been used to enable many different cryptocurrencies. Unfortunately, existing public blockchains and smart contracts deployed on them may disclose sensitive information. Although there is some ongoing work that leverage advanced cryptography to address some of these sensitive information leakage issues, they require significant changes to existing and popular blockchains such as Ethereum and are usually computationally expensive. On the other hand, private blockchains have been proposed to allow more efficient and privacy-preserving data sharing among pre-approved group of nodes/participants. Although private blockchains address some of the privacy challenges by allowing sensitive data to be only seen by the select group of participants, they do not allow public accountability of transactions since transactions are approved by known set of users, and cannot be accessed publicly. Given these observations, one natural question that arise is, can we leverage both public and private blockchain infrastructures to enable efficient, privacy enhancing and accountable applications ? In this work, we try to address this challenge in the context of digital auctions.**

Mainly, we propose a novel hybrid blockchain architecture that combines private and public blockchains to allow sensitive bids to be opened on a private blockchain so that only the auctioneer can learn the bids, and no one else. At the same time, we leverage public blockchains to make the auction winner announcement, and payments accountable. Furthermore, using smart contracts deployed on public blockchain, we show how to incentivize truthful behavior among the auction participants. Our extensive empirical results show that this architecture is more efficient in terms of run time and monetary cost compared to pure public blockchain based auction implementations.

I. Introduction

Recently, blockchain technologies have been applied to many important problems ranging from crypto currencies to digital identity. By removing the need for having a centralized trusted entity for financial transactions, cryptocurrencies emerged as the early critical application of the blockchain technology. As blockchain technology evolved, two types of blockchains have emerged: Public/Permissionless blockchains and Private/Permissioned Blockchains. Public blockchains allow any participant to observe all the transactions and the data stored on the blockchain. Although, this is useful in some applications such as cryptocurrencies, it creates significant challenges in applications such as healthcare where the privacy of the data needs to be preserved. Recently, different solutions that try to address these privacy issues by leveraging advanced cryptographic techniques (e.g. [1], [2]) have emerged. Unfortunately, these solutions are either too expensive to support general smart contract execution (e.g., [2]) or require significant changes to underlying blockchains such as Ethereum (e.g., [1]). At the same time, private blockchains such as HyperLedger Fabric [3] have emerged to enable more efficient and privacy-preserving solutions for application scenarios where the participants are pre-screened. For example, multiple companies involved in a international trade transaction may create a private blockchain to streamline trade processing. In this scenario, a private blockchain may allow faster processing (e.g., no need for mining etc.) and increased privacy (i.e., only the participating parties can see the potentially sensitive data). Unfortunately, these private blockchains do not allow public accountability. For example, the public will have no way of seeing any transactions and/or associated monetary movements. This lack of public accountability could be an issue for certain applications. For example, with important government contract bidding, public accountability could increase public trust and potentially reduce fraud and nepotism. Ideally, solutions that preserve privacy, and enable public accountability without requiring significant changes to existing blockchains are needed.

In this work, we try to address some of the challenges discussed above using a novel solution that combines public and private blockchains to enable privacy-enhancing and accountable auctions. Our proposed solution uses standard cryptographic tools (e.g., bit commitments) on the public blockchain to keep necessary accountability information while limiting the amount of sensitive information disclosed. Private blockchain based smart contracts are used to securely find the auction winner; and finally, public blockchain based smart contracts are used to declare auction winner and punish any party that tries to cheat.

To our knowledge, this is the first work that combines public and private blockchains to enable privacy-enhancing and accountable auctions. The main contributions of this work can be summarized as follows:

- We propose a novel hybrid blockchain architecture for online auctions. The proposed architecture limits the disclosure of sensitive bid information and enables the public accountability of the final auction results.
- We empirically show that the proposed hybrid architecture enables more computationally efficient and less expensive (e.g., it uses a reduced amount of gas on the public blockchain such as Ethereum) auctions as compared to pure public blockchain based auction solution.
- We show that the proposed solution only discloses potentially sensitive bid information to the party that is running the auction. This in return reduces the private information leakage. Furthermore, we show that any cheating party could be *monetarily punished using public smart contracts*.

The rest of the paper is organized as follows: In section II, we discuss the related work. In section III, we provide the overall system architecture. In section IV, we discuss the smart contracts used for the proposed solution. In section VI, we analyze the security and privacy guarantees provided by our system. In Section VII, we provide the detailed experimental evaluation of the proposed system. Finally, we conclude by the future work discussion in Section VIII.

II. Related Work

There has been a significant amount of research towards adding cryptographic techniques to a distributed ledger. The blockchain has also been evaluated and redesigned architecturally to either make the ledger more secure or commit transactions more faster. There is also an important effort on developing multi-ledger applications. In this section, we describe previously proposed solutions and discuss how our proposed architecture fulfills some outstanding challenges and helps make the blockchain a faster and more secure infrastructure for auctioning.

A. Enhancing Blockchains with Cryptography

One possible method of preserving privacy is by adding cryptography in the smart contracts, which are a medium of transacting data between distinct parties. Privacy preserving smart contracts have been introduced in the Hawk [2] paper, which talks about a blockchain model of cryptography and transactional privacy. It leverages zero-knowledge proofs to achieve this goal. However, it is inefficient because every time a transaction is generated, it has to pass through the cryptographic primitives, which take significantly longer to mine in the real world.

Ma et al construct non-interactive zero knowledge(NIZK) proofs [4] with the help of homomorphic encryption to ensure validity of "private" transactions. These proofs attempt to hide the balance and transaction amount of any transaction by creating time-efficient generated NIZK proofs, but at the expense of lengthier proofs.

Zkledger [5] is another approach that protects privacy of participants by using cryptographic techniques to achieve complete auditing in an efficient manner using rolling caches. It eliminates the requirement of a third party auditor who is able to access all transactions.

Zerocash [6] is a system that assists the bitcoin network to use zero-knowledge proofs to ensure the privacy of a transaction.

Unlike these efforts, we use efficient and simple one-way permutation bit commitment protocol and standard encryption techniques already used in existing blockchains.

B. Blockchain with Architectural Modifications

The recently proposed Ekiden [1] is another platform for confidentiality-preserving, trustworthy and high performance smart contracts. It tackles the issue of lack of confidentiality and poor performance over existing ledgers by combining blockchain with their custom trusted execution environment (TEE) based on the Intel SGX architecture. However, it is exposed to a plethora of attack vectors due to the fact that it is heavily relies on a TEE, which proves to be a bottleneck. One such attack is when the TEE and the blockchain are out of sync, an attacker can easily forge blocks and invalidate genuine transactions.

Recently, doing off-chain computation emerged as an important direction to improve the efficiency of the public blockchains. For example, the bitcoin lightning network [7] attempts to solve the problem of delayed transaction confirmations over a public ledger, by introducing a network of micropayment channels or transaction channels that occur off-blockchain. It addresses the issue of security challenges related to off-chain transactions by using a new signature hash to allow transaction malleability and public broadcast in the event of an uncooperative participant with the help of timelocks.

We choose a combination of smart contracts and the Ethereum Virtual Machine to build a private infrastructure to enhance the performance of our system.

C. Multi-ledger Blockchain Applications

Xinfin [8] focuses on creating a hybrid blockchain with the help of Ethereum's quorum architecture that builds private channels over the public blockchain. They fork the quorum blockchain to improve the throughput of transactions and create a light weight client that connects natively to the system. However, a drawback of the Xinfin architecture is that it puts in a lot of trust in third parties like the network manager, smart contract manager and the administrative manager.

Enigma proposes another approach to addressing privacy challenges [9]. It allows transactions over a peer-to-peer network while the data is kept private. An external blockchain is used to control this network to manage access control, identities and tamper-proof log of events. Security deposits, fees and incentivized operations help prove the correctness and fairness of the system. With the help of this system, users are able to share their personal data using cryptography.

Secure auctioning has been considered in Strain [10]. However, similar to earlier research, they focus on using zero-knowledge proofs to perform privacy preserving auction with improved efficiency. They attempt to be efficient with low blockchain latency using a weak adversary model.

Auction, being an ideal use-case that require distributed parties to compete in the form of bids to win an item uses the Ethereum Virtual Machine deployed on a combination of private and public infrastructure.

D. Comparison with Our Approach

To the best of our knowledge, *this is the first work that leverages public and private blockchains to enable privacy enhancing (e.g., disclosure of bidding information is limited) and publicly accountable auctions* without using expensive cryptographic tools or trusted execution environments.

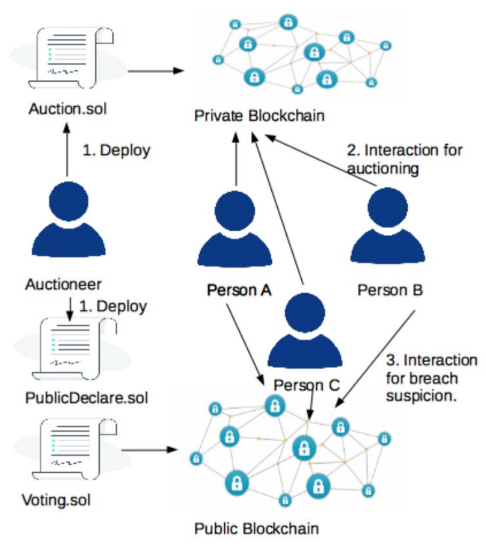


Fig. 1: **Architecture of the proposed system:**

III. System Architecture

Our proposed hybrid auction system encompasses two blockchains - a private blockchain and a public blockchain, and three types of actors participating in the auction as shown in Figure 1. The actors include an auctioneer, multiple participants of the auction and voters who vote when a contract breach [1] is suspected. The private blockchain is hosted by the auctioneer. The auctioneer also acts as the mediator or a liaison between the private and the public blockchains. The auctioneer posts the results of the auction conducted on the private chain to the public chain. The participants in the auction can also join in as voters in case a breach is suspected by any participant in the system. The voters also verify the information posted by the auctioneer on the public chain is correct. As we discuss later on, this voting mechanism can be used to punish potential cheaters as well as incentivize truthful behavior.

[1]We use the term *breach* to describe any situation that deviates from the truthful execution of the auction.

The private blockchain is instantiated using system parameters that do not cause any delay in the bidding process. For example, we assume pre-funded accounts for established participants to avoid delays caused due to mining when starting an auction. The difficulty is also set to a very low value to allow for faster mining of transactions. The private and the public blockchains are interlinked with the help of a module that opens sockets to both chains. The sockets initialize a connection between the chains effectively creating a layer of communication between the two chains.

The private blockchain is a quasi-permissioned blockchain. This means that there is no trusted third party or a membership service provider that provides membership to join the blockchain. However, the auctioneer acts as an entity that is able to permit a proposed participant to join the network. The decision is made over the private chain with the consensus of other participants. Once the decision is made and the request is approved, the access control list on the private chain is updated and the auctioneer opens a connection with the interested participant by sending the network connection details to the future participant, thus encouraging the participant to connect to the network. The interested participant uses these details to join the network. Once on the network, the newly joined participant can start making bids using the deployed auction smart contract on the private chain.

Each actor in the system communicates with both the private and public chain. Since both chains are exclusive, each actor holds a separate account for each chain. Thus, each actor will have two accounts. The account address on the public blockchain is not configurable and is generated dynamically when the account is created. However, in a private blockchain, the account address is predetermined by the auctioneer who is initializing the chain. A hash table in the form of a key-value pair is used for the storage of multiple account address corresponding to the same actor. The participants confirmed to participate in the auction first have to be known to the auctioneer so that the hash table containing records of participants along with their addresses is updated. Whenever the participant decides to leave the system, the hash table entry with the corresponding address is deleted, effectively disabling the participant from interacting with the smart contracts on both chains.

A. Smart Contracts

There are three smart contracts deployed in order to enable our architecture - the auction contract, the PublicDeclare contract and the CongressFactory contract [11].

The auction contract is deployed on the private blockchain by the auctioneer. The public blockchain hosts two smart contracts namely the PublicDeclare contract and the Congress-Factory contract. The former is used to declare the highest and second highest bidders along with their respective bids of the auction. The latter is used to generate a Congress contract in case a breach occurs. The declared winner of the auction has to be consistently agreed among all participants. If that does not happen, then a Congress contract is created using the CongressFactory. This starts a voting mechanism between the participants to check certain parameters whether the declared winner is indeed true. There are multiple methods implemented to validate the result posted by the auctioneer. One of them is by checking commitment values. In order for the auction to be secure, the participants bid on the items in the form of commitments. During the bidding phase the CongressFactory is used to record all the bids in the form of commitments. When a breach is suspected, the CongressFactory has an automated check that requires the auctioneer to send its data to it. The CongressFactory validates it with the data it possesses, subsequently confirming whether the auctioneer has committed a breach.

Each participant deposits ether into the CongressFactory to join the hybrid chain. In order to claim a breach on the hybrid network, the accuser must submit a deposit to the CongressFactory. Only when a deposit is paid, will the CongressFactory start a Congress contract for voting purposes. The reason being, if the suspected breach is false, then the money deposited by the accuser will be divided between the CongressFactory and the accused participant. This discourages false accusations and encourages true reporting. If the breach reported is indeed true, then the accused loses all its money in the CongressFactory to the rest of the participants and unless he/she does not deposit more money into the CongressFactory, the breacher is forced to forfeit his/her status as an auction participant and is deleted from the system.

A sealed bid auction is performed with the help of commitments generated offline by each participant. While the participants are conducting bids in the form of commitments, a timer is running constantly in the background to set an upper limit for the amount of time allotted to the item dispatched for auction. If the timer runs out the auctioneer sees this as a form of auction completion and initiates the process of declaring the winner on the public blockchain. In order to declare the winner on the public blockchain, the auctioneer starts a ”Reveal phase” that queries the participants for the values and random numbers used to generate the commitments. The auctioneer verifies whether the values used to generate the commitments are consistent with the generated commitments. Once consistency is verified, winner is declared, else the violator of the auctioning process is penalized.

B. Commitment Protocol

The bidding phase is divided into 2 distinct phases: the commit phase and the reveal phase. The auctioneer notifies each participant via private chain about the beginning and ending of the bidding and reveal phases. When each participant receives the notification, they can choose to bid and participate in the auction. For security and privacy reasons, each participant uses the modified one way permutation bit commitment protocol to encapsulate their bid into a hash value. This hash value is recorded on the public chain due to its immutability property. When the auctioneer ends the bidding phase and starts the reveal phase, each participant who has sent a bid to the public chain, must send a revelation, which include the value and the random number used, to the auctioneer over the private chain. The auctioneer, after reveal phase, will retrieve all the commitments made by each participant from the public ledger and will automatically check and penalize participants who have not sent their revelations or who have sent false revelations with the help of both the private and public smart contracts. The winner (highest bidder) and second highest bidder of the auction will subsequently be declared over the public chain.

There are potentially two auctioneers in the system. At an instance, only one auctioneer acts as the primary auctioneer. The other auctioneer is secondary in stand-by mode. This helps in maintaining a fault-tolerant environment. The secondary auctioneer can become the primary auctioneer in multiple cases. (1) The primary auctioneer malfunctions. (2) The primary auctioneer is voted as a breacher by the voters on the public chain.

IV. Smart Contracts for Secure Auctions on Hybrid Blockchains

In this section, we discuss the three smart contracts deployed to enable secure auctions on hybrid blockhcain architecture - the auction contract, the PublicDeclare contract and the CongressFactory contract.

A. Auction Smart Contract

The auction smart contract is deployed by the auctioneer. At the time of deployment, the contract stores the address of the auctioneer. The auctioneer sets the time limit the contract will last when the contract is deployed. This time limit is modifiable by the auctioneer only. This time limit once set, will remain the same for each auction session until it is modified for a new auction session. Once the auctioneer determines the auctioning item, the participants start calling the bid function on the public chain to make a bid (Algorithm 3: Lines 1 - 4)and during the Reveal phase, the bid commitment will be injected in the private chain (Algorithm 1: Lines 10 - 14) to verify the revealed commitments match what the participants have bid on the public chain. This bid function is called only by the auctioneer. This is accomplished by adding the modifier to the contract in the form of a ‘require’ function which throws an exception if the above condition is not satisfied.

Once every participant has made a bid on the item being auctioned, the reveal phase starts. This phase of the bit-commitment scheme involves the auctioneer querying for randomness r and bid value m from every participant (Algorithm 1: Lines 15 - 27). Each participant sends the value m as a payment via a payable function of the smart contract and the random number r for the auctioneer to confirm whether the revelation of the commitment is indeed true.

When the reveal phase is ended by the auctioneer, the winner and the second highest winners along with their bids are set (Algorithm 1: Lines 28 - 39) to be *declared on the public ledger*.

Algorithm 1 PRIVATE AUCTION

```
1:  procedure AUCTION CONTRACT
1:  StartBid
2:      require (msg.sender = auctioneer)
3:      $bidStarted \leftarrow true$
4:  EndBid
5:  StartReveal
6:      require (msg.sender = auctioneer)
7:      $bidStarted \leftarrow false$
8:      $revealStarted \leftarrow true$
9:  EndReveal
10: Start makecommitment (bidder,bid)
11:     require (msg.sender = auctioneer)
12:     require (revealStarted = true)
13:     $bids(bidder).push \leftarrow bid$
14: End
15: Revealcommitment (values(),randoms())
16:     require (revealStarted = true)
17:     $bidLength \leftarrow bids(msg.sender).length$
18:     require (bidLength = values().length)
19:     require (bidLength = randoms().length)
20:     for $k \leftarrow 1$ to $values().length$ do
21:         if $bid(msg.sender)[k] = BITCOMMITMENT(values(k))$ then
22:             $trueAmount \leftarrow values(k)$
23:         else
24:             $falseAmount \leftarrow falseAmount + values(k)$
25:             $EMIT\ PENALIZE\ EVENT$
26:     $checkIfWinner(msg.sender, trueAmount)$
27: End
28: CheckifWinner (bidder,value)
29:     if $value <= highestBid$ then
30:         if $value >= secondHighestBid$ then
31:             $secondHighestBid \leftarrow value$
32:             $secondHighestBidder \leftarrow bidder$ return true
33:     else
34:         $secondHighestBid \leftarrow highestBid$
35:         $highestBid \leftarrow value$
36:         $secondHighestBidder \leftarrow highestBidder$
37:         $highestBidder \leftarrow bidder$
38:         $EMIT\ BID\ INCREASED\ EVENT$
39: End
```

If the participant wins, an event triggered causes the auctioneer to check the balance in the smart contract and declares the winner on the public chain. As shown in table II, the auction ends when the timer runs out, subsequently triggering an event that the intermediary application linking public and private blockchains is listening for. Once the application captures this event, the control is transferred over to the public blockchain by the auctioneer. Once a new item is declared to be auctioned (Table II), the auctioneer can start the bidding phase again and the cycle repeats.

Triggered By	Triggered For	Purpose
Auctioneer, auction finalized	All	Winner shall now pay the amount to the owner
Participant, Breach Detected	Auctioneer	Auctioneer will reveal commitments to Participant
Participant, Participant on Breach	Auctioneer	Voting Contract Triggered
Voting Contract	All	Results will be declared

TABLE I: Voting Events

Triggered By	Triggered For	Purpose
Participant, A commitment is made	Auctioneer	Auctioneer registers and publishes to all
Auctioneer, Auction End	Participants	Reveal Phase starts Participants no longer able to perform auction
Auctioneer, New Item	Participants	Commit Phase starts Participants can perform auction

TABLE II: Auction Events

B. AuctionPublic Smart Contract

On the approval of the auctioneer, this smart contract (Algorithm 2) declares the winner with the second highest bidder on the public blockchain once the auction is ended on the private chain. The auctioneer declares the winner and the second highest bidder (Algorithm 2: (Lines 1 - 7)) of the auction on the public chain and that triggers another event that provides an appropriate notification to each participant regarding the details of the highest bidder and the second highest bidder. The winner of the auction is provided notification to pay the second highest bid amount declared on the private blockchain to the smart contract on the public blockchain. Once the bid amount is paid on the public chain, another event triggers the smart contract to pay the bid amount to the auctioneer effectively completing the payment on the public blockchain and closing the auction. The other participants are provided details of the sealed bid auction performed, also known as **Vickrey Auction** [12]. These participants may have the choice to protest this claim in case they suspect some malicious activity (Algorithm 2: (Lines 12 - 15)).

C. Voting Contract

The CongressFactory contract (Algorithm 3) is called by anyone who suspects the breach. In order to join the hybrid chain system, each participant must deposit some pre-decided cryptocurrency amount. It is also used as the contract that stores bids made during the bidding phase (Algorithm 3: Lines 1 - 4).

This contract spawns a new Congress contract depending on the parameters passed by the participant as shown in Table I. In conjunction to creation of a new contract, the CongressFactory automatically checks if the auctioneer is at fault by retrieving the revelation from the auctioneer and comparing the bid commitment values stored with the values generated from the revelations by the auctioneer (Algorithm 3: Lines 8 - 17). This

Algorithm 2 PUBLIC DECLARE

```
procedure PUBLIC DECLARE CONTRACT
setHighestBid (bid1,bid2)
    require (msg.sender != auctioneer)
    $bids(bidder).push \leftarrow bid$
End
setHighestBidder (bidder1,bidder2)
    $commitments(accused) = auctioneerCommitments(accused)$
End
getHighestBid () return highestBid1, highestBid2
End
getHighestBidder ()return highestBidder1, highestBidder2
End
suspectBreach (accused())
    $congressObject \leftarrow CongressFactory(Address)$
    $breachContract \leftarrow congressObject.createCongress(accused, msg.sender)$
End
```

new contract will work separately for separate auction items or auction sessions. For example, if participant A suspects a breach on item A, then a contract will be called specifically for item A. To determine whether a breach has occurred or not, lies in the hands of a panel of jurors decided by the accuser. In the case the accuser is unsure of whom to include in the jury, the participant can let the contract decide the set of jurors. A default set of jurors pre-defined by the CongressFactory contract can be injected into the newly created smart contract. The jurors will vote on the basis of certain contractual clauses resembling an actual legal contract. Each juror will return a decision whether a breach has occurred or not. A quorum of jurors will then be chosen randomly and declare the result. Depending on the result, the participant will either lose his/her committed amount or will be paid a reward and make the auction happen again. Once the results are declared, instead of having the Congress contract lie idle in the blockchain ecosystem, after a certain time limit, it will destroy itself effectively not allowing any participant to call the spawned contract again.

D. Integrating Smart Contracts

The auctioning process is orchestrated by the auctioneer. The auctioneer deploys the contracts on the hybrid blockchain. The private chain's auction is executed first. Once deployed, the intermediary application is notified and the auctioneer can then start the 2 phase bid. The timer is constantly running in the background and will end the auction if it runs out, thus triggering an event to transfer control back to the application. The winner, the runner-up and their respective bid amount is decided by the auctioneer by automated checks in the smart contract. The application then prompts the auctioneer to call the public blockchain to declare the winner and the runner-up.

Algorithm 3 PUBLIC VOTING

```
procedure CONGRESS FACTORY CONTRACT
Start makecommitment (bidder,bid)
    require (msg.sender != auctioneer)
    $bids(bidder).push \leftarrow bid$
End
setCommitmentFromAuctioneer (auctioneerCommitments,accused)
    $commitments(accused) = auctioneerCommitments(accused)$
End
autoCheckAuctioneer (accused)
    $breachSuspected \leftarrow false$
    for $k \leftarrow 1$ to $commitment(accused).length$ do
        if $commitments(accused)[k] = auctioneerCommitments(accused)[k]$ then
            $breachSuspected \leftarrow false$
        else
            $breachSuspected \leftarrow true$ return
    return breachSuspected
End
autoCheckCommitment (accuser,revelation)
    if $commitments(accuser) = BITCOMMITMENT(revelation)$ then
        if $winningBid < revelation$ then
            PenalizeAuctioneer
            $breachSuspected \leftarrow true$ return
        else
            $breachSuspected \leftarrow false$
            PenalizeAccuser return

    else
        PenalizeAccuser return
End
createCongress (accused(),accuser)
    $congressContract \leftarrow new\ Congress(bidders)$
End
```

Once the rankings are declared, the smart contract waits for the winning participant to make a payment of the second highest bid to itself. The payment is then made to the auctioneer on the confirmation of the above successful transaction. While the PublicDeclare contract is still active on the public chain, every other participant on the chain have the right to suspect a breach. The party that breach the protocol can be the winning participant or the auctioneer. In case the auctioneer is held guilty, then the secondary auctioneer conducts future auctions. Once the penalty is applied, the designated auctioneer repeats the cycle for another auction. Thus, the flow of the system is maintained making it secure with commitments, and more efficient with the help of a private blockchain. This efficiency gain is due to running certain parts of the overall system in the private blockchain. See section VII for more details. Whenever a breach is suspected, the CongressFactory automatically checks whether the auctioneer has committed a breach. This is done with the help of the bid commitments made by each participant in the bidding phase. The bit commitment of the highest Bidder is derived from the revelation on the public chain is compared with the bitcommitment stored by the CongressFactory. If they match, then the bit commitment of the accuser on the CongressFactory is compared with the bit commitment generated by the revelation of the accuser. If they match and the revelation sent by the accuser is higher than the winning bid, then the auctioneer is penalized, since that is a breach committed by the auctioneer.

V. Implementation

For the implementation of our hybrid architecture, we have chosen Ethereum for both the private and public chains.

A. Platform Setup Details

The Ethereum Virtual Machine is used for both the private and public blockchains. The private blockchain is set up in a virtual private cloud over multiple hosts taking advantage of the Amazon Web Services (AWS) Elastic Compute Cloud (EC2) instances. Port 30303 is exposed for the nodes to discover each other. Ropsten [13] is used as the public blockchain.

B. Private Chain Specifications

1) The Genesis Block: A genesis file for the private blockchain is written in json format. This file is a customized unique genesis file that contains details regarding the gas limit, the chain id, difficulty, nonce and most importantly the participating account ids with pre-funded balances. The genesis block is the first block in the blockchain or block 0 and is created using this genesis state file. The gas limit is the highest amount of gas that can be used to execute any transaction on the private chain. This is set to a high value, a standard 2100000 to avoid being limited when the auction is being conducted since as the auction progresses, the value being transacted can get large eventually throwing an error if the gas limit is set too low. The difficulty is set to a low value to avoid wastage of time while waiting for transactions to get mined. The goal here is to make computations on the private chain as fast as possible by removing any delays due to the mining of transactions.

Every participant address is pre-funded by 100,000 gas for initial bidding purposes. The auctioneer is provided 1,000,000 gas. The auctioneer provides each new participant joining the private chain with 50,000 gas to make initial transactions. Once the participant joins the system, it is enforced to become a miner.

2) Commitments over Private Chain: The commitments are generated offline by the participants by a JavaScript application portal. The participants then with the help of the web3 application send the generated commitments to the auctioneer. The web3 application makes a call to the method in the auction contract deployed on the private chain to perform a sealed bid auction.

3) Miners over the Private Chain: Commitments [14] are mined as transactions by the miners over the private blockchain. The set of miners is restricted to the participants for block creation. Each participant is enforced to become a miner on the private chain. In order to have each participant act as a miner, each participant should have their account address configured to be a miner over the private chain. However, since the private chain has no real "ether", each participant has no incentive to become a miner since that would waste their computational resource and therefore, the enforcement. Also, when a participant joins the private chain, it does not have enough gas to participate in the bidding process. It first has to become a miner and after a considerable amount of time, it's account starts accumulating enough gas to participate in bidding. Thus, we establish the following parameters in the system: (1) each participant is pre-funded with a necessary amount of ether to participate in transaction. (2) each participant has to become a miner on its acceptance to the private chain.

4) Ether over Private Chain: Although the ether on the private chain has no significant effect on the economic ecosystem of the blockchain, every entity on the private blockchain should be provided with equal amount of ether as every other entity. If an imbalance occurs, the wealthier entity can transact repeatedly to halt the auction. Thus, the amount of gas provided should be equal among all.

C. Mechanism of Formation of a Bid Commitment

The bit commitment scheme [14] implemented using a one-way hash function scheme. Let f be a one-way hash function with hard-core predicate B. Let m be the binary form of the bid. Each participant randomly selects r which is a string of 0's and 1's. The bid function accepts commitments in the form of (1) $y = f(r)$ and (2) $b = B(r) \oplus m$. The bid is parsed before acceptance by the contract. If there is any discrepancy in the bid e.g. the commitment does not match the required length or it contains any invalid characters, the bid is rejected.

D. Integrating Public and Private Blockchains

Nodejs [15], a JavaScript run-time environment and an npm package manager is being used for the front end UI for the participants and the auctioneer to call the blockchain. The nodes can also use the command line to interact with the blockchain. The geth [16] command is used to connect to the blockchain from the respective nodes. The Nodejs application have API calls to both the private and the public blockchain.

Using truffle [17] is the approach taken to simulate the connections to the private and public blockchains simultaneously in the event of Nodejs failure. Testrpc [18] is used as a testing framework for the private chain. The smart contracts being deployed on the blockchains are written in Solidity [19] and compiled in the remix browser [20]. Remix is used to deploy the smart contracts on the public blockchain. Metamask [21] sends a browser based notification to the user for approval when a transacation on the public chain is made by the Nodejs application. Web3js [22] is the API being used to call smart contract functions over the private and public blockchains.

VI. Security and Privacy Analysis

In this section, we give an overview of the security and privacy guarantees provided by the proposed system.

A. Security Analysis

In our setting, we can easily prove that if the auctioneer cheats in disclosing the winner (assuming that the auctioneer can arbitrarily break ties among bidders who bid exactly the same amount), then a malicious auctioneer will be detected using the public smart contract.

Theorem 1. *Assuming that the auctioneer and the highest bidder do not collude and that the public smart contract works correctly, then any cheating by the auctioneer in declaring an incorrect winner can be automatically detected.*

Proof. Please note that all participants commit their bids to the public blockchain at the beginning of the protocol. If for any reason, the auctioneer declares some other party P and the winning price p as the winner, then the correct winner P' who bid p' such that $p' > p$, can prove to the smart contract that $p' > p$ by opening its commitment using the appropriate smart contract function. Therefore, the auctioneer's cheating can be automatically detected. □

The above theorem states that if there is no collusion, and public commitments of bids to smart contract works correctly, the auctioneer's cheating in declaring the winner can easily detected. Unfortunately, this observation would not be correct in announcing the correct second price bid. The auctioneer can always claim that the second price bid is $p-\epsilon$ where p is the highest bid. The above theorem and the proof reasoning directly apply if the auctioneer discloses not only the winner but simultaneously the second highest bidder as well. This way if the second highest bidder does not collude with the auctioneer, then this cheating can be easily detected using the same detection mechanism and the auctioneer can be automatically penalized. Still, even a colluding second highest bidder cannot force the winner to pay more than this winning bid p.

B. Privacy Analysis

In our system, by design, only the auctioneer will see all the bids. The other participants only see the winner's address, the winner's bid price, the second highest bid price, and the second highest bidder's address.

In some cases, disclosing all the bids even to the auctioneer can be considered a privacy violation. We believe that using ideas similar to Ekiden [1] *just for the private blockchain* [2] could be considered. We leave such an extension as a future work.

[2]Please note that this would not require any changes to public blockchain such as Ethereum and will still be much cheaper computationally since it is used just on the private blockchain

VII. Experimental Evaluation

We conducted extensive experimental evaluation to understand the performance of the system. More specifically, experiments were conducted to measure the running time and gas consumption (i.e., monetary cost) for every transaction while varying the system parameters and deployment scenarios. We discuss these experiments and the results below.

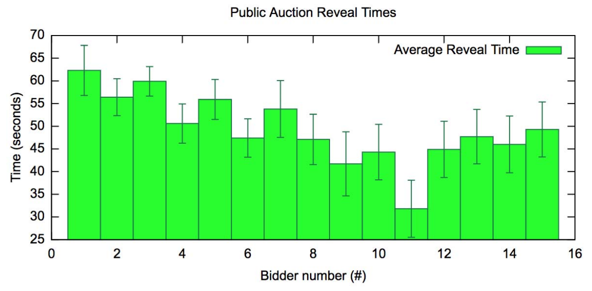
A. Hybrid Architecture vs. All Public Architecture

The *Hybrid Architecture* has auction on the private chain, and declaration, voting on the public chain. The *All Public Architecture* has auction, declaration and voting on the public chain, eliminating the private chain. We have deployed the auction contract along with the remaining public contracts on the public blockchain and then compared the time and gas recorded for one round of a complete auction with the respective performance results recorded on a hybrid auction. The results are shown in the graphs above. The time and gas consumed on the hybrid architecture is insignificant since it uses a private blockchain isolated from the external world. However, the public architecture records a significant amount of time for each transaction in Figure 2a because all transactions happen on the public blockchain. When the corresponding auction transaction is recorded on the private blockchain in the hybrid architecture, the time decreases significantly as shown in Figure 2b. The cumulative gas consumed by the public architecture is shown in Figure 3a and Figure 3b. Figure 3a records the gas consumed by each initial bid and consecutive bids made made by each individual bidder as they start bidding in the system. The initial bid made by each bidder is large due to the change in system state as a new bidder is added to the ledger in the form of an account address. Figure 3b shows that the amount of gas taken in the reveal phase is directly proportional to the number of bids revealed. Thus, we can deduce that the cumulative gas consumed on the public architecture is much higher than the gas consumed on the hybrid system since there is no real ether consumed by the transactions on the private chain.

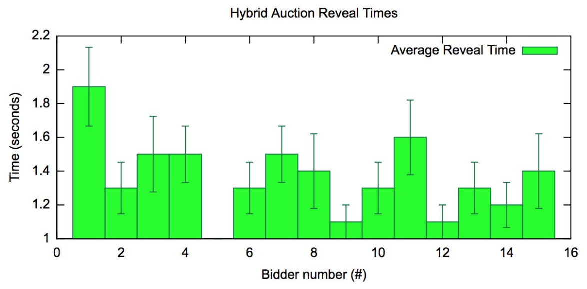
Thus, from the results shown, we can say that the hybrid blockchain based auction system is significantly faster and cheaper than an all public blockchain based auction system. We have also analyzed the number of transactions that are confirmed in time t on the hybrid chain vs. the all public chain. For each transaction confirmed on the public chain, 50-60 transactions are confirmed on the private part of the hybrid chain. Thus, effectively reducing the time by at least 2000 % and increasing the efficiency by 5000%. Here, we are disregarding the CongressFactory smart contract in both architectures since in both the Congress contracts are deployed on the public blockchain and would show similar performance and gas consumption.

B. Bid Commitment & Breach Detection: Gas Analysis

During the reveal phase, the bids to the CongressFactory of the participants were recorded. For each participant, the first bid made has an increased cost but consecutive bids

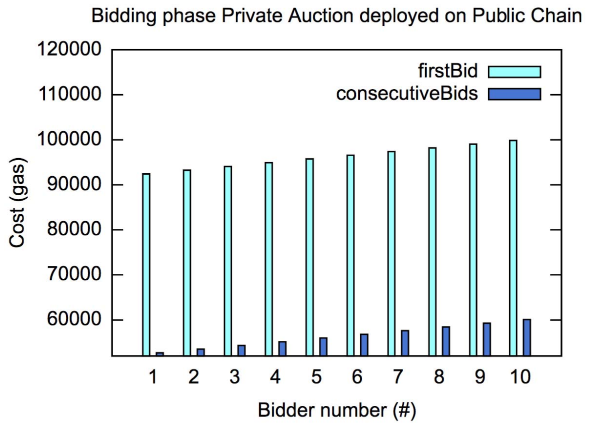


(a) **Public auction reveal time**

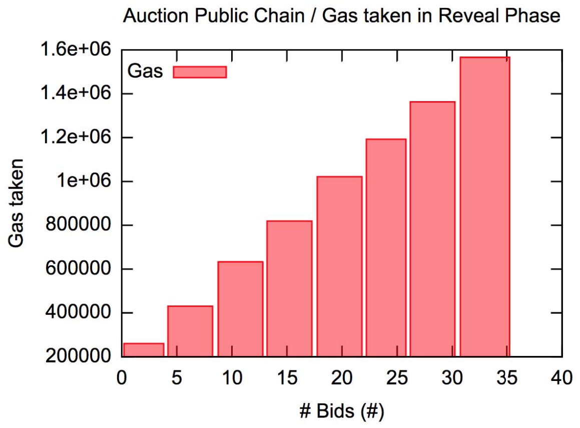


(b) **Hybrid auction reveal time**

Fig. 2: *Experimental results of Time Comparison between a Public auction and Hybrid auction.*



(a) **Bidding Phase Private Auction deployed on Public Chain**



(b) **Reveal Phase Private Auction deployed on Public Chain**

Fig. 3: *Experimental results of Gas Taken of the Auction Contract if the entire system is deployed on the Public Chain.(Note: It would be cost-free on the Hybrid Chain)*

have a constant cost. The reason is that when a new bidder is added to the system, the system's database, in our case hashmaps, are updated with the information of the new bidder that includes the participant's public address. On subsequent bids by the same bidder, since the bidder's information is already populated in the system, the state of the contract does not change with any new information besides the value of the bid commitment itself. Thus, as seen in the Fig. 4a, there are 2 data points for each bidder. The first bar corresponds to the gas used when the first bid is made. The second bar corresponds to the gas used when consecutive bids are made.

On suspecting a breach on any auction actor by any other actor (participant / auctioneer), the larger the number of accusers, the more gas it takes to complete the breach decision. When an auction ends, if a participant suspects a breach, a new Congress contract is created. As shown in the Fig. 4b, it takes 1.02e+06 gas if only one participant suspects a breach. If multiple participants suspect a breach, the gas used to spin up each Congress contract increases linearly. As an example, if ten accusers suspect a breach, the total gas spent on making a decision on a single breach increases up to 1.09e+06.

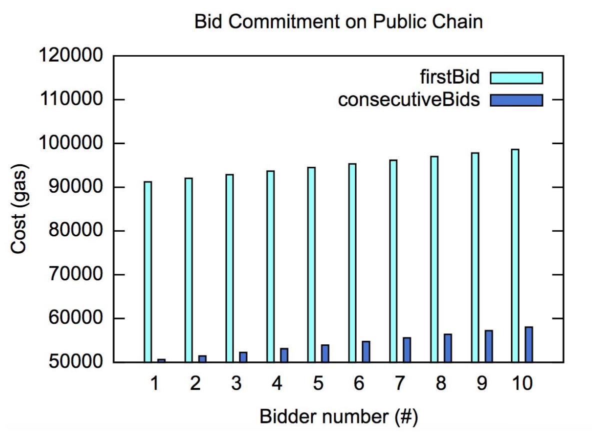
We have analyzed the CongressFactory in our previous work [11]. It is a very efficient means to inspect breaches since it enables the creation and destruction of child contracts in parallel to other auctions being conducted.

Our experimental results show that (i) the hybrid blockchain based auction is cheaper and faster than an all public blockchain auction architecture, and (ii) the hybrid blockchain based auction architecture provides enhanced privacy.

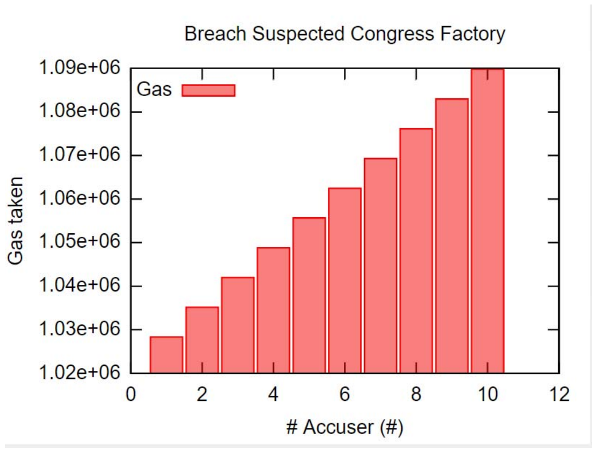
VIII. Conclusion

With blockchain technology gaining importance as a distributed computing infrastructure, privacy becomes an important challenge that needs to be addressed. Current research in this space focuses on using advanced cryptography, which may be computationally expensive. We propose a novel hybrid architecture that enables privacy-enhanced, accountable auctions where sensitive bids are opened on a private blockchain and public blockchains are use for payments and accountability.

This hybrid blockchain architecture can be used for a wide range of applications apart from auctions such as digital assets exchanges, sharing personal identifiable information (PII), real-time data data record, and notarization. For digital assets exchanges, the payment transactions can be conducted



(a) **Bid Commitment on Public Chain**



(b) **CongressFactory Breach Suspected**

Fig. 4: *Experimental results of Gas Taken of the Public Contracts.*

on the public chain and the exchange of digital assets can be conducted on the private chain enabling the information about ownership and exchange of specific assets to remain private. Our hybrid approach can also be used to share PII. Public details like first name, last name, ethnicity, etc can be shared with entities such as Govt. Officials on the public chain whereas private details like Social Security Number(SSN), address, and Passport number can be shared over the private chain. Since the receiver of the PII will host the hybrid blockchain, the combination of private chain details and the public chain details can be validated against the official database and if there is a mismatch, then the sender's account will be penalized enforcing accountability. The hybrid blockchain can also be used as a notary service, having official documents signed and sent by the "trusted" addresses over the private chain and the payment / confirmation being made over the public chain.

As part of our future work, we plan to create a framework for multiple blockchains hosting different applications and show how we can better preserve privacy when private, public and consortium blockchains are combined. We also plan to address the challenge of multi-currency exchange thus making our platform cryptocurrency agnostic.

Acknowledgments

The research reported herein was supported in part by NIH award 1R01HG006844, NSF award CICI-1547324, ARO award W911NF-17-1-0356 and MITRE under the MIT SuperUROP program.

References

[1] R. Cheng, F. Zhang, J. Kos, W. He, N. Hynes, N. Johnson, A. Juels, A. Miller, and D. Song, "Ekiden: A platform for confidentiality-preserving, trustworthy, and performant smart contract execution," 2018.
[2] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "The blockchain model of cryptography and privacy-preserving smart contracts," in *IEEE Symposium on Security and Privacy (IEEE SP)*, 2016.
[3] "Heperledger." https://www.hyperledger.org/, 2018. [Online].
[4] S. Ma, Y. Deng, D. He, J. Zhang, and X. Xie, "An efficient nizk scheme for privacy-preserving transactions over account-model blockchain," in *Cryptology ePrint Archive, Report 2017/1239*, 2017. https://eprint.iacr.org/2017/1239.
[5] N. Narula, W. Vasquez, and M. Virza, "zkledger: Privacy-preserving auditing for distributed ledgers," in *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18), USENIX Association*, 2018.
[6] E. B. Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza, "Zerocash: Decentralized anonymous payments from bitcoin," in *IEEE Symposium on Security and Privacy*, 2014.
[7] J. Poon and T. Dryja, "The bitcoin lightning network: Scalable off-chain instant payments." http://lightning.network/lightning-network-paper.pdf, 2015.
[8] X. organization, "The xdc protocol technical whitepaper." www.xinfin.org, 2017.
[9] G. Zyskind, O. Nathan, and A. Pentland, "Enigma: Decentralized computation platform with guaranteed privacy." www.enigma.co.
[10] E.-O. Blass and F. Kerschbaum, "Strain: A secure auction for blockchains," in *European Symposium on Research in Computer Security, pages 87–110. Springer*, 2018.
[11] Harsh Desai, Kevin Liu, Murat Kantarcioglu and Lalana Kagal, "Adjudicating violations in data sharing agreements using smart contracts," in *The 1st IEEE International Conference on Blockchain (Blockchain-2018)*, 2018.
[12] D. Lucking-Reiley, "Vickrey auctions in practice: From nineteenth-century philately to twenty-first-century e-commerce," *Journal of Economic Perspectives—Volume 14, Number 3—Summer 2000—Pages 183–192*, 2000.
[13] "Ropsten testnet pow chain." https://github.com/ethereum/ropsten, 2018. [Online].
[14] O. Goldreich, "Foundations of cryptography." Vol 1, Vol 2, 2004.
[15] "Node.js." https://nodejs.org/en/, 2018. [Online].
[16] "Go-ethereum." https://github.com/ethereum/go-ethereum, 2018. [Online].
[17] "Truffle." https://github.com/trufflesuite/truffle, 2018. [Online].
[18] "Ganache-cli." https://github.com/trufflesuite/ganache-cli, 2018. [Online].
[19] "Solidity." https://solidity.readthedocs.io/en/v0.4.21/, 2018. [Online].
[20] "Remix." https://remix.readthedocs.io/en/latest/, 2018. [Online].
[21] "Metamask." https://github.com/MetaMask, 2018. [Online].
[22] "Web3js." https://github.com/ethereum/web3.js, 2018. [Online].