MaRSChain: Framework for a Fair Manuscript Review System Based on Permissioned Blockchain

Nitesh Emmadi
TCS Innovation Labs, India
nitesh.e@atc.tcs.com

Lakshmi Padmaja Maddali
TCS Innovation Labs, India
padmaja@atc.tcs.com

Sumanta Sarkar
TCS Innovation Labs, India
sumanta@atc.tcs.com

ABSTRACT

Current Manuscript Review Systems (Conference/Journal) rely on a centralized service (like EasyChair, iChair or EDAS) which manage the whole process that starts with manuscript submissions to notification of the results. As the current review systems are centralized, the trust is based on a single entity which controls such a system. The fairness of the system hinges on the honesty of the central controlling authority. This dependency can be avoided by decentralizing the source of the trust. Other related problems are to detect double(plagiarized) and concurrent submissions, i.e., submission of a manuscript to multiple forums concurrently, during the review period. There are some third party services like iThenticate which can detect plagiarized contents. However, there is no mechanism available in the current manuscript review systems which can detect concurrent submissions. The cryptocurrency Bitcoin has shown the power of decentralization and shared database through blockchain technology, and currently is being studied for its immense impact on FinTech. We leverage blockchain to address the above concerns and present a manuscript review system that provides trust, fairness and detects double and concurrent submissions. As a proof of concept, we developed a prototype of our MaRSChain system on top of Hyperledger Fabric platform. To the best of our knowledge, this is the first ever decentralized manuscript review system based on Blockchain.

KEYWORDS

Manuscript Review System, Blockchain, Consensus, Fairness, Double submission, Concurrent submision, Smart Contract

1 INTRODUCTION

Ever since Bitcoin [15] showed the application of blockchain technology that rules out the need of central authority, it has drawn attention from both the industry and academia. Blockchain enables mutually distrusting parties form a peer to peer distributed network and maintain a common transaction ledger. A typical blockchain as used in Bitcoin does not need verified identity of a peer, i.e., it is an open enrollment system. In other words it is a permission-less blockchain. Research is being carried out to embrace blockchain's decentralization feature in several application ranging from finance [18], supply chain [16], IoT [20] and to many other business use cases[21] [17] [7]. Note that the idea of permission-less blockchain may not be suitable for many enterprise applications, like banks, which require their users to be verified. To cater to this kind of application we need a permissioned model of blockchain, for example, Hyperledger [10] by Linux Foundation. With this permissioned blockchain many centralized services can be decentralized which is being now explored. In this paper we focus on the applicability of permissioned blockchain to build a decentralized conference management system.

A conference management system (for example, EasyChair [8], EDAS, [9] or iChair [11])[1] handles the life-cycle of a conference from manuscript submissions to acceptance/rejection notification. A conference organizing committee, first invites manuscripts for reviews and assigns the manuscripts to reviewers for evaluation. Based on the evaluations submitted by the reviewers, the conference program chairs decide on the manuscript acceptance. The accepted papers are invited for publication in the conference proceedings.

The current conference management systems are centralized services, thereby giving the hosting entity full control of the system. A malicious party in control of the system can manipulate decisions and results impacting fairness of the system. For instance, the controlling entity can assign papers to reviews of his choice and hamper the fairness of reviews. It is even possible for the controlling party to change the results in the system. Thus, there is need for a fair decentralized service which inhibits malicious parties from corrupting the system. We propose a blockchain based conference management system which does not rely on any central service or platform.

Another important concern with the current review systems is detection of double and concurrent submissions [19]. Submission of plagiarized content to a forum is referred to as *d*ouble submission and submission of a manuscript to multiple forums concurrently is referred to as *c*oncurrent submission. It is against the interest of research community to tolerate double and concurrent submissions as the reviewers efforts put into the review process are wasted. Forums advise the manuscript authors against double and concurrent submissions (see ASIACCS submission guidelines [3]). The current systems rely on a third party services like iThenticate [12], to detect double submissions. Therefore, double submissions can only be detected effectively as long as this third party is honest. Moreover, content of a submitted manuscript in a conference is not public, thus it is not possible to check whether the same manuscript is not submitted in two concurrent forums. This is very similar to the double spending problem in Bitcoin where the platform has to restrict users from transacting the same bitcoin concurrently in two different transactions. Cryptocurrencies handle double spending

[1] EasyChair and EDAS are third party services whereas iChair is an open software that can be hosted by any of the program chairs of a forum. Note that these are centralized services.

by enforcing a common ledger for all the miners to check against. It is interesting to note that the problem of double and concurrent submissions is pretty similar to the double spending problem in Bitcoin. In this regard, we propose a decentralized solution based on blockchain for detecting double and concurrent submission as in cryptocurrencies. We propose enforcing a shared ledger for all the forums to detect concurrent submissions. Indeed our system is a collection of blockchains.

1.1 Related Work

Other related work known in the literature includes ConfiChair[2] and P3ERS[1]. ConfiChair proposes an architecture to build conference management system in a privacy preserving manner in order to protect against untrusted cloud service providers. Confichair preserves privacy and confidentiality using encryption mechanisms with key translations and mixes. P3ERS (Privacy Preserving Peer Review System) is a distributed peer review system with several group managers. P3ERS preserves privacy of all the users in the system with improved group signature scheme. P3ERS considers untrusted cloud service provider and actors within the system as potential adversaries. P3ERS proposes a distrubuted architecture to host different services on different servers eliminating a single point of attack. However, these systems are stand-alone conference management systems and can not detect double and concurrent submissions.

1.2 Our Contribution

In this paper, we propose MaRSChain, a framework to build manuscript review system based on permissioned blockchain, that guarantees fairness and detects double and concurrent submissions.
We briefly outline how a permissioned model differs from a permission-less model of blockchain and describe why a permissioned blockchain is suitable to build MaRSChain in Sec.[2]. We leverage Hyperledger Fabric [10], a permissioned blockchain platform, to build our system. Fabric provides several interesting features and help realize the objectives of MaRSChain. MaRSChain can be built on top of any permissioned blockchain platform which provide features described in this paper. Some of the features we desire for our system include cryptographically pseudonymous one-time identities for users, confidentiality of transactions, Smart Contracts [7].

Smart Contract is an encapsulation of domain logic executed by nodes of the blockchain. In MaRSChain, Smart Contracts are designed to handle submissions and reviews, validation of submissions and consolidation of reviews.
MaRSChain promises:

- Security against manipulation of manuscript reviewers assignment
- Security against manipulation of manuscript reviews
- Double and Concurrent submission detection
- Anonymity of authors and reviewers
- Confidentiality and Privacy of manuscript submissions and reviews

2 PERMISSIONED VS PERMISSION-LESS BLOCKCHAINS

Blockchain is an immutable shared ledger which enables distrusting parties to engage with each other. The identities of parties can broadly be classified as verified or unverified and this is the core criteria determining the choice of blockchain model (permissioned/permission-less). A permissioned model of blockchain is more suitable in case of enterprise applications where the distrusting parties involved in business have verified identities. Whereas permission-less blockchains are for applications with anonymous unverified parties [14].

A conference review system is an application operating in controlled environment that employs parties with verified identities i.e., authors, reviewers, program chairs. Conference review systems may require pseudonymous submission of manuscripts and reviews by authors and reviewers respectively. Permissioned blockchains like Hyperledger [10] provide auditable pseudonymous identities for users to transact anonymously on the ledger. Permissioned blockchains allow authorized parties to link pseudonymous identities to original identities. Hence, a permissioned model of blockchain platform that provides such features suffice for MaRSChain.

3 SYSTEM OVERVIEW

MaRSChain is a network of multiple mutually segregated permissioned blockchains, we call individual blockchains as *c*hannels. These blockchains can be classified as:

- **Conference Blockchains(CBC)**: individual forums maintain independent blockchain systems with their respective program chairs as nodes. The CBCs have a list of papers submitted to the respective forums.
- **Publishing House Blockchain(PHBC)**: a blockchain network with publishing houses as nodes. PHBC has a list of publications already published in all forums and the list of manuscripts under-review at all forums. It is reasonable to assume several(or all) publishing houses to be part of a single blockchain network as it is in their mutual interest and interest of their clients to detect double submissions. Also storing data on the blockchain defends their ownership over the data and will be helpful in case of copyright issues too.

We first describe a single conference blockchain(CBC) *c*hannel and then show how individual conference blockchain *c*hannels together with publishing houses form a publishing house blockchain(PHBC) *c*hannel.

3.1 Conference Blockchain(CBC)

3.1.1 Entities in CBC. The entities in a CBC are listed below:

- **Authors:** Users submitting manuscripts to the conference are authors. Authors are end-users in the CBCs.

- **Reviewers:** Reviewers review manuscripts submitted by the authors. Reviewers are also end-users in the CBCs.
- **Program Chair(PC):** Program Chairs are responsible for manuscript validation, reviewers assignment and updating final manuscripts status (proceedings) to PHBC. PCs are the nodes in the CBCs.

A conference blockchain can be visulalized as in Fig.[1].

3.1.2 Operational Flow. We describe the operational flow of our system below in sequence (as in Fig.[3]). All the transactions in MaRSChain are listed in G.

Manuscript Submission. All the inputs to the blockchain systems are signed transactions which are recorded in the blockchain. Authors submit their manuscripts to a conference as signed transactions (CBC_Submit_Txns). On receiving the CBC_Submit_Txns, the PC nodes in the CBCs validates the submissions for:

- **Policy Validation:** Semantics, signatures and submission policy.

Smart Contract 1: Manuscript submission to forum

Transaction: CBC_Submit_Txn=(*manuscript*, *author pseudonyms*)
Validation: Check if manuscript is already submitted to this forum
if *manuscript is present in the ledger* **then**
  duplicate submission; *rejected*
else
  considered for submission; *under_submission*
end

- **Double and Concurrent Submission Detection:** The conference PC queries the PHBC to check if the is double or concurrent submission. If yes, PC rejects the submission, else the conference accepts the submission for review, and propagates the paper information to PHBC (PHBC Submit Txns). A detailed description of double and concurrent submission validation is provided in PHBC Sec.[3.2].

Smart Contract 2: Double/Concurrent Submission Detection

Transaction: PHBC_Submit_Txn=(*manuscript*, *program_chair_id*)
Validation: Check if manuscript is submitted to any of the forums; Query manuscript status from PHBC
if *manuscript already present in PHBC* **then**
  if *manuscript_status is "published" then*
    double submission;
    status updated to "rejected" in CBC
  else if *manuscript_status is "under − review" then*
    concurrent submission;
    status updated to "rejected" in CBC
  else
    earlier rejected submission; *accepted for review*
    status updated to "submitted" in CBC;
     status updated to "under − review" in PHBC
else
  new submission;
  status updated to "submitted" in CBC;
   status updated to "under_review" in PHBC
end

Reviewers Assignment. Once the manuscript acceptance window is closed, the PCs reach consensus on assigning the submitted manuscripts to reviewers for evaluation. For ease of implementation, we assume that this consensus is an offline process and all the PCs have agreed to the decision on reviewers assignment. A PC initiates the reviewer assignment transaction($CBC_Reviewer_Assignment_Txn$s) based on their decision for each of the manuscripts. All these transactions are validated by the PCs and their consensus reflects the reviewers assignments.

Smart Contract 3: Reviewers Assignment

Transaction: CBC_Update_ReviewersList_Txn=(*manuscript_id*, *reviewers_list*)
Validation: Check if reviewers list is acceptable(as decided by program committee)
if *manuscript reviewers list is acceptable* **then**
  update reviewers list for manuscript in CBC
else
  reject transaction
end

Review Submission. Reviewers evaluate the assigned manuscripts and submit their evaluations to the system in the form of transactions (CBC_Review_Txns). The reviewers evaluation transactions are validated by the nodes and the evaluations are updated in the system. An evaluation is a score given to the manuscript by the reviewer along with reviewer comments.

Smart Contract 4: Review submission to forum

Transaction: CBC_Reviewer_Txn=(*manuscript_id*, *review_pseudonym_id*, *review*, *score*)
Validation: Check if review is submitted by assigned reviewer
if *reviewer in reviewers list* **then**
  update review and score for manuscript
else
  reject transaction
end

Results Declaration. On receiving all the evaluations, the system smart contract consolidates the evaluation scores of each of the submissions and updates the results of the submissions. These scores reflect the decision of the conference. These scores are then notified to the authors.

Also, the PHBC has to be updated about the results of the

Smart Contract 5: Consolidate reviews of manuscripts

Transaction: CBC_Reviews_Consolidate_Txn=(*manuscripts*)
Validation: Check if manuscript status is "under-review"
for *all reviewedmanuscripts in CBC* **do**
  paper_final_score = (reviewer1_score + reviewer2_score + reviewer3_score)/3
  update manuscript final score in CBC;
  if *paper_final_score>3* **then**
    status updated to "accepted" in CBC
  else
    status updated to "rejected" in CBC
  end
end

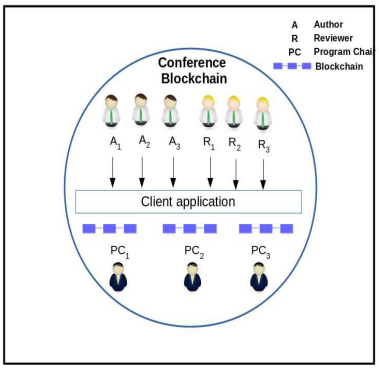

Figure 1: Conference Blockchain(CBC)

manuscripts. This is done through manuscript results transaction ($PHBC\_Manuscript\_Result\_Txn$) submitted by the PCs to PHBC. PHBC nodes update the status of each of the papers in the system.

Smart Contract 6: Update final status in PHBC

Transaction: PHBC_Update_Status_Txn=(*manuscripts*)
Validation: Check manuscript status for each of the reviewed manuscript
for *all reviewed manuscripts in CBC* **do**
  if *manuscript_status is "accepted" in PHBC* **then**
    status updated to "accepted" in PHBC
  else
    status updated to "rejected" in PHBC
  end
end

3.2 Publishing House Blockchain(PHBC)

3.2.1 Entities in PHBC:. The entities in a PHBC are listed below:

- **Program Chair(PC):** Program Chairs of several conferences(CBCs) are end-users in the PHBC. PCs query the PHBC for double and concurrent submission validation and are responsible for listing their manuscripts in PHBC blockchain.
- **Publishing House(PH):** Publishing Houses index and publish the proceedings (accepted manuscripts). PHs are nodes in PHBC.

Fig.[2] depicts architecture of MaRSChain system.

3.2.2 Operational Flow: The forums query the PHBC blockchain to detect double and concurrent submissions. By bridging different forums on a single blockchain together with publishing houses, we detect double and concurrent submissions without relying on a trusted third party service.

Double submission detection. In the current conference management systems, the forums rely on trusted third party services to detect double submissions. The third party has access to publication content from several different publishers. The trusted party performs a "Similarity Check" [6] to check for plagiarism of the submissions i.e., double submission detection. A submission is compared against public domain data and a database of current and archived publication content from publishing houses to check if the publication violates plagiarism. Usually, these services are provided by trusted third parties. For example iThenticate [12] is such a service, to whom all the member parties(users/publishers) contribute their full archived and current publication content. Therefore, double submissions can only be detected effectively as long as this third party is honest.

To solve this problem of trusting a third party to detect double submissions, we introduce another *c*hannel of blockchain with the publishing houses as nodes. This *c*hannel of blockchain maintains a ledger of all publication content from all the publishing houses. This content also includes the publications currently under review at all the conferences associated with the member publishing houses. Hence, this is a database of all the publications published in the past and the publications currently under review at several conferences. The program chairs of the conferences are users in this blockchain channel who can query and update the publications on this blockchain.

Whenever a publication is submitted to any conference, the program chairs query the publishing houses blockchain to know if that particular publication is already listed in the PHBC ledger as published. We assume availability of plagiarism software to detect double submissions when a publication is searched against a database of publications[2]. If the publication is already listed in the PHBC as published then it is a double submission. In this way, double submissions

[2]A simple plagiarism software tool can check if a similar manuscript is already listed in the system. It is also possible to leverage machine learning models to detect complex plagiarism violations.

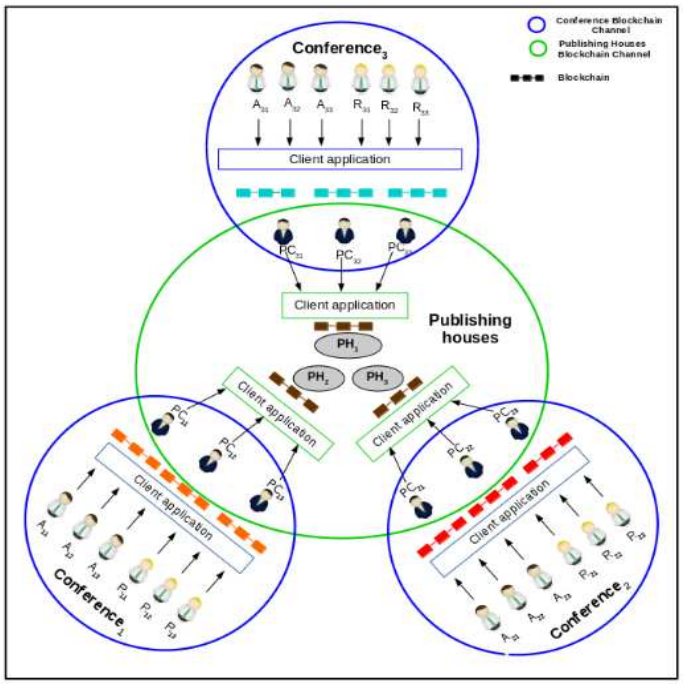

Figure 2: MaRSChain: A Manuscript Review System based on Blockchain

can be detected effectively. In case the submission is genuine, then a new entry is made into the blockchain ledger with status as under-review. This enables other forums to detect concurrent submissions of the same publication. Also, once the results of the conference are finalized, the status of the publications has to be updated by the program chairs in the PHBC.

Concurrent submission detection. In the current conference management systems, the conferences do not have access to other conference systems in order to detect concurrent submissions. In MaRSChain, the PHBC blockchain enables publishing houses to share a common ledger with forums. This blockchain ledger is a collection of all the publications published in the past and the publications currently under review at several conferences. Whenever a publication is submitted to any conference, the program chairs query the publishing houses blockchain to know if that particular publication is already under-review at a different conference and hence detect concurrent submissions. Therefore, by registering all the submissions on the blockchain, it is possible for the conferences to detect concurrent submissions.

4 SECURITY OF OUR SYSTEM

All the security guarantees in MaRSChain are established intuitively from well studied primitives. In this section, we describe how our system guarantees security against several adversarial models.

4.1 Security against Double and Concurrent Submissions

Double and Concurrent submission detection is an important feature of our system. A double or concurrent submission can be submitted either by a malicious author alone or by a malicious author in collusion with a malicious PC.

MaRSChain ensures detection of a double and concurrent submission.

Proof Sketch: Recall that, all the publishing houses maintain a list of all manuscripts submitted to all the conferences. This list includes manuscripts already published at the previous conferences and the manuscripts currently under review at all the conferences.

- **Double and concurrent submissions by malicious author:** Consider a malicious author submitting a double-submission through a submit transaction to the forum. This submit transaction is validated by the PCs in the forum to check if the submission qualifies to be reviewed. As part of the validation the manuscript is validated for double submission through a query to PHBC blockchain. The PHBC easily detects the double submission and responds appropriately to the PCs. The PCs can then reject the submission.
- **Double and concurrent submissions by malicious author in collusion with malicious PC:** Conference PCs on receiving a submission, validates them against this list of publications by querying the

publishing houses blockchain and detects double submissions. This validation is part of the submission validation in the conference blockchain and is done by all the PCs and the submission is accepted on consensus of the PCs. Hence, even a malicious PC can not manipulate the system to accept a double submission.

Secure *consensus* of the PCs ensures rejection of double submissions.

4.2 Security against manipulation of reviewers assignment

In a centralized system, a malicious PC can affect the fairness of the reviews by assigning the manuscripts to reviewers of his choice in order to get his desired evaluation result.

MaRSChain provides protection against manipulation of reviewers assignment.

Proof Sketch: Consider a malicious PC submits a reviewers assignment transaction with the manipulated reviewers assignment to the system. The transaction submitted by malicious PC goes through consensus where each of the other PCs validate the transaction. A malicious assignment can easily detected by the other PCs and the transaction is rejected in the consensus. This ensures protection against unfair reviewers assignment. The consensus algorithm ensures that a malicious PC can not influence reviewers assignment at his will without corrupting enough other PCs. Honest majority of the PCs ensures that the system remains fair.

4.3 Security against manipulation of submitted reviews

We assume a malicious PC member trying to manipulate review evaluations to impact the decision on a manuscript. This can be done either by forging review transactions of the reviewers or by corrupting all the PC nodes to exclude certain review transactions.

MaRSChain protects users against forgery or system corruption to ensure fair reviews.

Proof Sketch: Consider a malicious PC member as an adversary trying to manipulate the review of a manuscript. To do so, the adversary has to do one of the following:

- **Forge signatures of reviewers:** To manipulate the review result of a manuscript, the PC member has to affect the reviews submitted by the reviewers. The reviewers submit a review transaction digitally signed[13] by reviewer's private key which is securely stored at his end. Security of a digital signature scheme ensures that a signature can not be forged without the corresponding private key. To forge a review, the adversary has to break the signature scheme. Hence, protection against forgery is ensured.
- **Exclude reviews from ledger:** Excluding the review of a manuscript from the ledger either by denying the reviewer transaction or by re-writing the ledger is one way for the adversary to manipulate the review. Decentralized system ensures that a single malicious party can not influence the system i.e., all the agreements are mutual through *consensus*. A consensus algorithm requires at least 't' out of 'n' parties in the system to be malicious in order to influence the system. Therefore, the system remains secure as long as the number of malicious PCs is less than 't'. 't' varies between one consensus algorithm to another. A permissioned blockchain platform like Fabric, considers several consensus algorithms like Practical Byzantine Fault Tolerance (PBFT) [5], Solo, SIEVE [4], etc.

The digital signature schemes and secure consensus mechanisms ensure protection against malicious PCs. Hence, security against manipulation of reviews is guaranteed.

4.4 Privacy of Authors and Reviewers

All the users in the blockchain network submit transactions to the system which are validated and recorded into a common ledger. Since the ledger is visible to all the entities in the network there is a concern for privacy of the users. A user can see all the transactions from other users in the system. In case of MaRSChain, an author can monitor transactions from other authors or reviewers and their review assignments. For fair reviews, the forums encourage anonymous submissions and blind reviews. Lack of information about the authors limits unwarranted behavior of reviewers in evaluation of manuscripts. A permissioned blockchain platform provides an anonymization mechanism to conceal the identities of the users on the ledger. One-time pseudonymous identities are provided to the users by the platform.

MaRSChain ensures privacy of authors and reviewers.

Proof Sketch: Privacy of the users in blockchain can be viewed in two forms: Anonymity and Unlinkability. Anonymity of transaction refers to hiding of the a user's identity in an anonymity set of all the users i.e., an identity on the ledger should not directly be associated to particular user. Unlinkability refers to the association of multiple transactions of a single user i.e., two different transactions from a same user should not be related to each other. To realize anonymity and unlinkability, pseudonymous identities are required for the users. All the transactions submitted to the blockchain are under pseudonymous identities. Thus, a transaction on the blockchain ledger can not be linked to a user directly. Pseudonyms also provides protection against linkability of transactions i.e., multiple transactions of same user. Only authorized parties (nodes) with the knowledge of a secret have the ability to link pseudonymous identity on the blockchain to the actual identity of the user. Therefore MaRSChain guarantees privacy by enabling the authors and reviewers to transact anonymously on the blockchain.

4.5 Confidentiality of submissions

As blockchain is a shared ledger i.e., all the entities in the network have access to the blockchain ledger. All the transactions (submission and reviews) in MaRSChain are stored on the ledger. It is essential to maintain confidentiality of the transactions in the interest of authors and reviewers. To enable confidentiality of the transactions, a permissioned blockchain supports encryption of transaction payloads with one-time symmetric keys and access control mechanisms.

MaRSChain ensures confidentiality of transactions.

Proof Sketch: In a permissioned blockchain, the users are enrolled in to the blockchain network through an identity management service. During the enrollment, each user is provided with a secret keys to enable encryption of confidential transactions. These secret keys are long-term keys and are stored securely on the user-end. When a user wishes to submit a transaction to the blockchain platform, the user derives a one-time symmetric key (per transaction secret key) from the long-term key with the help of a Key Derivation Function (KDF). The transaction being submitted is encrypted with this one-time symmetric key. The symmetric keys are only available to the users themselves and other authorized parties in the network (nodes). Hence, an unauthorized adversary can not decrypt the transactions without the knowledge of secret key. Furthermore, access-control mechanisms that restrict access to transaction payloads provides another layer of security for the transactions. Thus, the confidentiality of the users' data is preserved.

5 IMPLEMENTATION

We developed a prototype of our MaRSChain system on top of Hyperledger Fabric platform. We simulate various steps of a conference management system to illustrate the practicality. We remark that a MaRSChain system can be built on top of any permissioned blockchain platform. Note that Hyperledger Fabric platform is still evolving and certain features like cross-channel communication are not yet available. Hence, we opted for off-the-band mechanism (off-chain script) to communicate information from one channel to the other. It can be seen that a full-fledged system can be built once the platform is fully developed. A detailed document describing the whole setup and other instructions is available along with the implementation. Our implementation does not use plagiarism software tool, instead we simply compare and match "paper_id" string of submissions. Our transaction structures are as described in G. Each of the blockchain channels in our implementation comprises of four nodes and the consensus requires majority of the nodes i.e., 3 out of 4 nodes, to endorse a transaction.

5.1 Bootstrapping PHBC

PHBC blockchain is bootstrapped with a list of manuscripts with various statuses. The statuses of the manuscripts are described in Table[1] . When a manuscript is queried by a forum program chair(PC), the status of a matching manuscript in PHBC is returned to the PC. The PC can then decide on the manuscript based on the status. This helps the conference PCs in detecting the double and concurrent submissions.

Manuscript Status	Description
submitted	already submitted to a forum and is under-review
accepted	already published at a forum
rejected	submitted earlier but rejected at a forum

Table 1: Manuscript status in PHBC

5.2 Conference Blockchains(CBCs) setup

Three individual conferences are instantiated with four program chairs each. Each conference submissions include all possible kinds of manuscripts i.e., new submissions, double submissions and previously rejected submissions. The statuses of the manuscript are described in Table[2] .

Manuscript Status	Description
under_submission	accepted by the forum for consideration
submitted	accepted for review by the forum after double/concurrent submission validation
under_review	assigned to the reviewers for evaluation
accepted	accepted by the forum for publishing
rejected	rejected by the forum

Table 2: Manuscript status in CBC

5.3 Double and Concurrent submissions Detection

The CBC submissions initially have status as "under-submission". These submissions are validated by CBC smart contract to detect double and concurrent submissions. The smart contract queries the PHBC with each of the manuscript to determine if the submission is a plagiarized manuscript or concurrently submitted manuscript. On detecting double or concurrent submission, the conference updates the statuses of the manuscripts accordingly. All the double and concurrent submissions are rejected by the conference and genuine submissions are accepted for evaluation.

5.4 Reviewers Assignment

Assigning reviewers to the manuscripts is done through consensus among all the PCs. Here, for ease of implementation we assume that the PCs discuss and decide on the reviewers assignment offline and submit reviewers assignment transactions based on this decision. Each manuscript is assigned to 3 reviewers. A reviewers assignment transaction, $CBC\_Reviewer\_Assignment\_Txn$, consists of *paper_id* and a list of reviewers assigned to that manuscript.

5.5 Review submissions

By the end of evaluation time-out, each reviewer submit review transactions to the CBC i.e., 3 reviews per manuscript. The PCs validate these transactions and update the

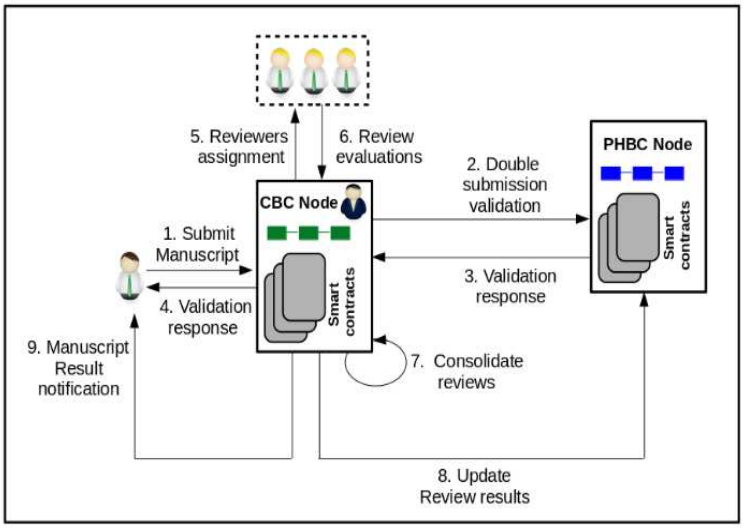

Figure 3: MaRSChain: Operational Flow

evaluations into the system. CBC_Review_Txn consists of $paper\_id$ and review evaluation(score and comments) of the reviewer.

5.6 Final Results

A smart contract is designed to consolidate the reviews of manuscripts and updates the results of manuscripts. We assume a simple consolidation of reviews by calculating average of the review scores given by the reviewers. Once the decisions on the manuscripts are finalized, results are notified to the authors.

Also, the PHBC is updated with the results of the manuscripts by CBCs through $PHBC\_Manuscript\_Result\_Txn$. Based on this transaction, the PHBC updates the status of the manuscripts to either "accepted" or "rejected".

5.7 Limitations

MaRSChain has subtle limitations in detection of concurrent submissions.

- **Concurrent submissions to different forums being validated at same instance:** There is a remote chance that a manuscript submitted to multiple forums being validated at an exact same instance of time thus evading double submission detection. However, this is very unlikely and can be handled by scheduling validations from different forums orderly without overlapping.
- **Concurrent genuine submissions from different authors:** It is also possible for two different authors submitting similar manuscripts to different forums honestly, leading to rejection of one of the submissions. This can be handled with manual intervention once the complete system is in place.

6 CONCLUSION

We have proposed framework to build a decentralized Manuscript Review System based on blockchain (MaRSChain) that ensures fairness and detects double and concurrent submissions. Our decentralized solution for a review system ensures integrity of data and helps honest authors, reviewers and program chairs. As part of our future work, we plan to integrate a reputation system to further strengthen MaRSChain system.

ACKNOWLEDGMENTS

We would like to thank Vigneswaran R for his inputs towards the development of our system.

REFERENCES

[1] Esma Aïmeur, Gilles Brassard, Sébastien Gambs, and David Schönfeld. 2012. P3ERS: Privacy-Preserving PEer Review System. (2012). http://dl.acm.org/citation.cfm?id=2423656.2423659
[2] Myrto Arapinis, Sergiu Bursuc, and Mark Ryan. 2012. Privacy Supporting Cloud Computing: ConfiChair, a Case Study. (2012).
[3] ASIACCS. 2017. (2017). http://asiaccs2018.org/?page_id=140
[4] M. Vukolic C. Cachin, S. Schubert. 2016. Non-determinism in Byzantine Fault-Tolerant Replication. In *International Conference on Principles of Distributed Systems(OPODIS)*. https://arxiv.org/abs/1603.07351
[5] M. Castro and B. Liskov. 1999. Practical Byzantine Fault Tolerance. In *Third Symposium on Operating Systems Design and Implementation (OSDI. USENIX)*. http://pmg.csail.mit.edu/papers/osdi99.pdf
[6] Similarity Check. 2017. https://www.crossref.org/services/similarity-check/. (2017).
[7] Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform. 2017. (2017). https://github.com/ethereum/wiki/wiki/White-Paper
[8] EasyChair. 2017. (2017). http://easychair.org/
[9] EDAS. 2017. (2017). https://edas.info/
[10] Hyperledger Fabric. 2017. https://www.hyperledger.org/projects/fabric. (2017).
[11] iChair. 2017. (2017). https://www.baigneres.net/ichair/
[12] iThenticate. 2017. (2017). http://www.ithenticate.com/

[13] Don Johnson, Alfred Menezes, and Scott Vanstone. 2001. The Elliptic Curve Digital Signature Algorithm (ECDSA). *International Journal of Information Security* (2001). https://doi.org/10.1007/s102070100002
[14] Arthur Gervais Karl Wst. 2017. Do you need a blockchain ? (2017). https://eprint.iacr.org/2017/375.pdf
[15] Satoshi Nakmoto. 2008. Bitcoin: A peer-to-peer electronic cash system. (2008). https://bitcoin.org/bitcoin.pdf
[16] Provenance. 2017. (2017). https://www.provenance.org/whitepaper
[17] RecordsKeeper. 2017. (2017). https://recordskeeper.co/
[18] Ripple. 2017. (2017). https://ripple.com/
[19] Henning Schulzrinne. 2009. Double Submissions: Publishing Misconduct or Just Effective Dissemination? *SIGCOMM Comput. Commun. Rev.* (2009). http://doi.acm.org/10.1145/1568613.1568622
[20] Hossein Shafagh, Lukas Burkhalter, Anwar Hithnawi, and Simon Duquennoy. 2017. Towards Blockchain-based Auditable Storage and Sharing of IoT Data *(CCSW '17)*. ACM. http://doi.acm.org/10.1145/3140649.3140656
[21] SimplyVitalHealth. 2017. https://www.simplyvitalhealth.com/. (2017).

G APPENDIX

Fabric supports chaincode execution through "*query*" and "*invoke*". A *query* is a chaincode execution which reads from the ledger but does not write into the ledger. Whereas, an *invoke* is capable of both, reading and writing. *invoke* transactions will be captured as transactions on blockchain. Here is the list of invoke and query transactions from our MaRSChain implementation:

Appendix G.A CBC Transactions

- ***C*BC_Submit_Txn:**
  peer chaincode invoke -o 127.0.0.1:7050 -C CBC_Channel -n CBC_CC -c '{"Args":["submit_papers", "author1", "author2", "author3", "attach_01"]}'
- ***C*BC_Update_Txn:**
  peer chaincode invoke -o 127.0.0.1:7050 -C CBC_Channel -n CBC_CC -c '{"Args":["update_paper", "paper_id", "status"]}'
- ***C*BC_Reviewer_Assignment_Txn:**
  peer chaincode invoke -o 127.0.0.1:7050 -C CBC_Channel -n CBC_CC -c '{"Args":["assign_reviewers", "paper_id", "reviewer1", "reviewer2", "reviewer3"]}'
- ***C*BC_Review_Txn:**
  peer chaincode invoke -o 127.0.0.1:7050 -C CBC_Channel -n CBC_CC -c '{"Args":["get_reviewers_decision", "reviewer1", "paper_id", "rating"]}'
- ***C*BC_Reviewes_Consolidate_Txn:**
  peer chaincode invoke -o 127.0.0.1:7050 -C CBC_Channel -n CBC_CC -c '{"Args":["make_decision"]}'
- ***C*BC_Query_Manuscript_Txn:**
  peer chaincode query -o 127.0.0.1:7050 -C CBC_Channel -n CBC_CC -c '{"Args": ["querySubmittedPaperInfo", "paper_id"]}'
- ***C*BC_Query_Manuscripts_List_Txn:**
  peer chaincode query -o 127.0.0.1:7050 -C CBC_Channel -n CBC_CC -c '{"Args":["queryAllPapersList"]}'
- ***C*BC_Query_Manuscript_Status_Txn:**
  peer chaincode query -o 127.0.0.1:7050 -C CBC_Channel -n CBC_CC -c '{"Args":["queryPaperStatus", "paper_id"]}'

Appendix G.B PHBC Transactions

- ***P*HBC_Submit_Txn:**
  peer chaincode invoke -o 127.0.0.1:7050 -C PHBC_Channel -n PHBC_CC -c '{"Args":["add_papers", "paper_id", "under_review", "pc_id_01"]}'
- ***P*HBC_Update_Txn:**
  peer chaincode invoke -o 127.0.0.1:7050 -C PHBC_Channel -n PHBC_CC -c '{"Args":["update_papers", "paper_id", "accepted"]}'
- ***P*HBC_Query_Manuscript_Txn:**
  peer chaincode query -o 127.0.0.1:7050 -C PHBC_Channel -n PHBC_CC -c '{"Args":["queryPaperInfo", "paper_id"]}'
- ***P*HBC_Query_Manuscripts_List_Txn:**
  peer chaincode query -o 127.0.0.1:7050 -C PHBC_Channel -n PHBC_CC -c '{"Args":["queryAllPapers"]}'
- ***P*HBC_Query_Manuscript_Status_Txn:**
  peer chaincode query -o 127.0.0.1:7050 -C PHBC_Channel -n PHBC_CC -c '{"Args":["queryPaperStatus", "paper_id"]}'