Reputation-Based Consensus for Blockchain Technology in Smart Grid

Lanqin Sang, Henry Hexmoor
Southern Illinois University at Carbondale

,

Abstract

Accurate and thorough measurement of all nodes' trustworthiness should create a safer network environment. We designed a blockchain-based, completely self-operated reputation system, R360, to enhance network security in decentralized network. R360 is a multi-factor measurement on the reputation of all nodes in the network. Specifically, a node's function, defense capability, quality of service provided, availability, malicious behavior, and resources are evaluated, providing a more accurate picture about a node's trustworthiness, which in turn should enhance the security of blockchain operations in a non-trust environment. Our design has been implemented on a single-machine, simulated setting, which demonstrates that, comparing to systems with few dimensions, R360's consensus took less time while at the same time providing better security to the system. Further security analysis shows that our design can defend common security attacks on reputation systems. Such a blockchain-based, self-operated reputation system could be used to manage smat grids for more efficient energy production and delivery.

Keywords: Smart Grid, Blockchain, Reputation-Based, Consensus, R360

1 Introduction

The development of society and technology demands revolutions in power generation, dissemination, and maintenance. As more and more suppliers and consumers are incorporated onto the power grid, the management and communication systems are increasingly complex, which calls for the transformation of old centralized systems into distributed systems, recently dubbed smart grid. Such distributed systems require novel software platforms for management.

Blockchain technology has been applied to many fields to manage distributed systems, such as future energy systems [1]. Blockchain is a distributed ledger system that collects blocks for registering different records of data or transactions. A key module of blockchain is consensus, which guards the system to produce accurate and identical information across the entire network. Blockchain consensus mechanism is a process for all nodes to agree or disagree on a set of transactions, also known as a block. Without the consensus mechanism, there will be no effective blockchain technology. However, current consensus mechanisms have their limitations, including low transaction throughput, weak consistency, and security concerns [2]. Vulnerabilities to attacks, double spending attacks, eclipse attacks, selfish attacks, and flash attacks are all examples [3].

Much research has been carried out to mitigate such limitations. The solutions either improve the low throughput with weak consistency or provide strong consistency but suffer liveness [4]. By and large, all the existing contemporary consensus mechanisms, such as proof-of-work, proof-of-stack and the variants based on them, rely on the assumption that most nodes are honest [5]. Although this assumption is reasonable, it is not based on facts. Recently, reputation-based consensus mechanism in blockchain has attracted attention. Proof of Reputation (PoR) is an upgraded, stronger, and more secure form of Proof of Authority (PoA) [6]. Proof of Reputation (PoR) consensus model depends on the reputation of the participants to keep the network secure. A node must have a reputation that is important such that they would face significant financial and brand value loss if they were to attempt to subvert the system.

Among all consensus strategies, PoR is most promising and deserves further investigation. The salient goal of consensus is to ascertain the data on the network are accurate and identical. If every node on the network is honest, there is no need for relying on a consensus mechanism. On the other hand, if the majority of the nodes on the network are not trustworthy, regardless of the amount of work or how much stake is invested, nothing will be truly meaningful. With reputation as the foundation, we would have know whether a node is trustworthy and how much you can trust it. Our research focuses on PoR. Current reputation-based designs are only based on a few dimensions of a node, such as blockchain functions, service quality, or computer power, and the nodes are treated differently (See Section 2 - Related Research). We propose to improve this situation in our design through a thorough reputation evaluation system, with all nodes having the same opportunities to various roles. In our model, a node's honest behaviors are rewarded and dishonest behaviors are punished. Nodes will be selected to publish blocks and vote commensurate with their reputation scores. According to [3], a reputation system must have the following three properties: (1) it must provide information that allows buyers to distinguish between trustworthy and non-trustworthy sellers; (2) it must encourage sellers to be trustworthy, and (3) it must discourage participation from those who are not trustworthy [7]. Thus, we propose the following design principles for our reputation system:

1. The basic properties of reputation. The concept of trust is always related to behavior or context[8]. A high level of reputation and trust obtained

in a context or through a behavior is not transferable to another context. For example, being a good doctor should not imply the person would be a good manager. The reputation of a person acting as a doctor or a manager needs to be treated separately. In the application to smart grid, a node's leadership reputation and voting reputation must be managed separately.

2. Fairness. A node's computation power, online time, etc. will be considered towards the node's reputation score. Time decay will also be considered.

3. Reputation-based qualifications and privileges. A node's qualifications and privileges to create blocks or to vote depend on its reputation scores.

4. Multidimensionality. We introduce the R360 concept, which encapsulates the notion that a node's reputation will be measured in multiple dimensions. According to [9], a reputation system includes the following fundamental dimensions: history, context, collection, representation, aggregation, entities, presence, governance, fabric, interoperability, control, evaluation, data filtering, and data aging, all of which have been used in reputation measurement. Based on these dimensions, we elaborate the design of our reputation system.

5. Two-tier block leader's and voter's selection. We first screen potential block leaders and voters by their total reputation to make sure they are good nodes in general. Then the leader will be selected by its block creation score and the remaining members of the consensus group will be selected by their voting scores. This strategy is extensively used as common selection principles in society, such as human employment practice.

The rest of paper is organized as following, Section 2 introduces related research, Section 3 presents our reputation-based model, Section 4 discusses the implementation and evaluation, and Section 5 outlines security analysis. We culminate the paper with concluding remarks.

2 Related Research

Classic consensus mechanisms include Proof of Work (PoW), Proof of Stake (PoS) and Practical Byzantine Fault Tolerance (PBFT) [10]. PoW is the first public blockchain consensus that was introduced in Bitcoin [11]. In this consensus mechanism, all nodes in the network are asked to solve a computationally expensive PoW problem. Whichever node that solves this puzzle first will be allowed to create a new block and other nodes can verify if this block is valid. PoW is vulnerable to 51% attack [10]; PoS [12] is the most popular alternative mechanism to PoW, whose goal is to overcome PoW's common limitations. In a PoS-based blockchain, mining a block is replaced by validating a block. The algorithm randomly selects validators to create new blocks and the probability of a node validating the next block is proportional to the stakes/assets it invests. Since the wealthiest validators administer the blockchain, this may introduce unfairness to the PoS consensus mechanism; Practical PBFT is a widely used mechanism. PBFT works on the assumption that at least 2/3 of the total number of nodes are honest [13].

Reputation-based consensus is quite different on how to select the block leader and consensus group members, how to produce reputation scores, and how to calculate or assign reputation scores. We outline a few of these systems next.

- RepuCoin [4]. The leader is randomly selected from the most reputable miners. The members of the consensus group are selected from the miners with the highest reputation scores. A miner's reputation score is based on the correctness of its behavior and is related to its computing power. The key block creator obtains a fixed mining fee and a share of the transactions according to its reputation. The selected leader takes the remaining transaction fees. The consensus group validates the blocks but receives no benefit from their work, which is unfair.

- Proof-of-Review [14]. The node with the most positive reviews will be selected as the round leader and can publish new blocks to earn more positive reviews. The nodes with the longest online time will be selected as consensus nodes. A node's trust value is based on the other node's reviews about previous transactions and interactions it had with other nodes.

- Proof of Reputation [15]. The node with the highest reputation score is selected as the leader to create a new block, the top 20% nodes in the reputation list are high-reputation nodes and participate in the consensus process. A node's reputation score is updated according to historical transactions, current age, participation in consensus, and illegal behavior.

- Proof of reputation [16] *et al.*: The node with the highest score will be the leader, which constructs a block and publishes it. Other nodes will verify that the block's sender is valid and consensus the block's transactions. A node's reputation is based on good behavior and block publication. At the end of each interaction, the service requester will generate a rating for the service and broadcast it. Other nodes will verify it and save the rating locally.

- Reputation-based neighborhood-watch mechanism [17] is used to detect and counteract the impact of data integrity attacks. The reputation is defined as trustworthy level that one controller could put on another controller. This design can detect colluding attacks that use false praise/false accusation. However, this mechanism would fail if multiple neighbors shared elaborate information to help each other to pass the information validation phase.

- Trust-management toolkit combines reputation-based trust with network-flow algorithms to identify and migrate faulty protection nodes[18]. The protection nodes monitor common power-grid variables and send them to the designated central node, which is selected from all the protection nodes. This design increases the robustness and lowers the risk of faulty node power outages. However, the central node selection and function defeats the basic purpose of blockchain.

- In R-CoDEMS framework, each agent monitors the correctness of received consensus estimation and assesses the reputation of its neighbors[19]. This design overcomes single and coordinated profit-driven attacks but cannot handle a local majority collusion.

- RBT is a distributed reputation system which is designed to improve blockchain consensus and peer-to-peer energy trading fairness[20]. The reputation is based on three roles: consensus node, energy seller, and energy buyer. However, this design only took the functions of nodes into consideration.

- A dynamic reputation–based consensus mechanism was proposed by Cai *et al.*[21]. In this design, a monitoring node selects consensus nodes according to their reputation ranking. Among the consensus nodes, the monitoring node randomly selects a primary node, which creates transactions and blocks. A node's reputation is based on its hardware memory, accessing system time, and storage performance. However, the functions of nodes were not considered.

In reported research, only a few dimensions have been used to evaluate nodes' reputation, whereas in some cases, some nodes are selected as judges/monitors to evaluate other nodes. Such designs are vulnerable and have inherent flaws. For example, a node that provides good services might have attacked other nodes. A node with much resources may launch selfish attacks. The process of monitor selection might be biased. Overall security can be greatly enhanced by adding additional security measures[22]. In this paper, we attempt to measure a node's reputation in multiple dimensions and provide all nodes with equal opportunities in order to overcome these flaws and potential biases. We name our reputation measurement system R360, in which we evaluate every node by its function, defense capability, offense, quality of service, availability, and resources. Our system also takes time decay into account. A node's behavior and ability determine its reputation scores, which in turn renders qualifications and privileges to the node. Penalties will be assessed for a node's incorrect decisions or inaction with free rides. Any serious offense will wipe out all of a node's positive reputation scores. If a node makes a correct decision on a block, it will be rewarded even if the block failed to validate. All nodes in the blockchain network form a self-managed dynamic system to provide a relatively trusted network environment.

3 System Model

3.1 System Composition

Whether a node is trustworthy or not depends on several factors. The R360 reputation system includes the following components, shown in Figure 1:

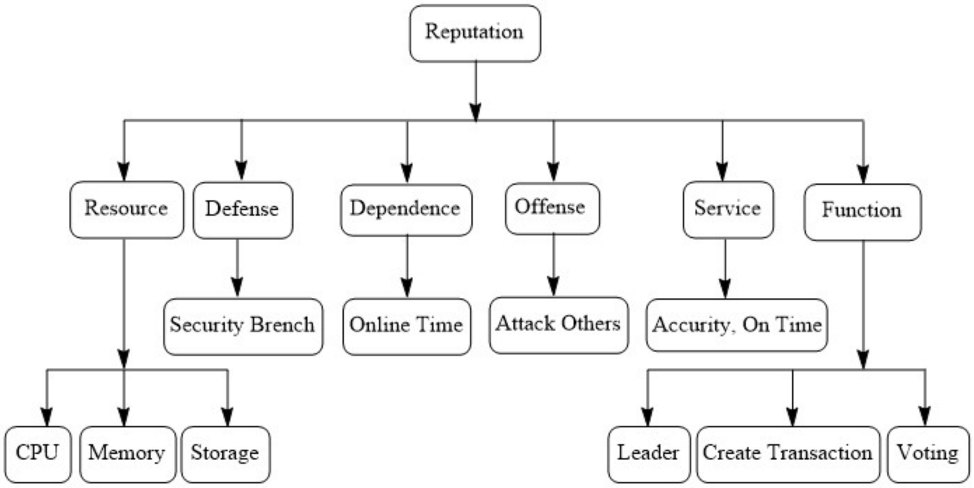

Figure 1: Reputation Structure

1. Resources. CPU, memory, and storage are necessary resources of any nodes and will seriously affect a node's performance.

2. Defense history. A secure node should be safe and free from security breach. A node that frequently or recently experienced security breaches will be less trustworthy.

3. Offenses. This score is related to a node's serious offensive behavior, which is not tolerated and subject to harsh punishment.

4. Function. In blockchain context, the core functions of any nodes include creating transactions, creating a new block and publishing it, and participating in consensus. A reputable node should function as expected.

5. Service. The service score of a node is given by those who receive its services. For example, the receiver can create transactions to evaluate the solar energy quality it received from a producer. It can rate if the information or assistance it obtained from a node is accurate and on time.

6. Dependence or Availability. This score measures a node's online time. A reliable node must be available to provide services when needed.

3.2 Reputation Score of Factors

1. Resource score. Resource score is not cumulative. There are four levels for the resource reputation score, 0, 1, 2 and 3. A node ranked at least 80% of the maximum of any specific resource, such as CPU, will be assigned 3 points, those ranked at least 60% will be assigned 2 points, those ranked at least 30% will be assigned 1 point, and the ones ranked lower than 30% will receive 0 point.

2. Defense capability score. Defense capability score is a cumulative measure. Based on the assessment of the damage caused by a security breach, the score can be -1, -2, -3, with -3 being the most serious value. All increments are added to the current defense score. According to [17], the impact of data breaches will likely diminish over time. In our design, we assume the impact of a breach will be completely erased after 5 years. Each year, 20% of defense score is reduced.

3. Service score. The service score is cumulative. After some nodes receive services, they will evaluate the service provider with a possible score of 0, 1, 2, or 3, depending on how well they rate the service they receive.

4. Function score. The function score is cumulative as a result of the work a node has performed in history. There are three kinds of function scores: transaction score for creating transactions, leading score for creating and publishing blocks, and voting score for participating consensus. Leading and voting scores are based on the number of transactions contained in the past block. If the number of transactions in a validated block is at least 80% of the highest number of transactions in a block in history, the voting score will be set to 3. If it is at least 50%, the voting score will be set to 2. If it is below 50%, the voting score will be set to 1. If there is no transaction in the block, the voting score will be 0. A voter will receive the voting score, and the leader will earn twice as many points. If a block failed, the voters who failed the block will receive the voting points, but the other voters and the leader of the block will receive the corresponding negative voting points; For transaction creation score, if the number of transactions is at least 80% of the maximum transactions created by a node in the validated block, the node that has created the transaction will be assigned 3 points, the nodes with a number of at least 50% will be assigned 2 points, and the nodes that created at least one transaction will be assigned 1 point; if a node didn't create any transactions, it will receive 0 points. If a block failed, the nodes that created transactions in the failed block will receive the corresponding negative score.

5. Availability score. The availability score is not cumulative, and it is only valid for one year. Ninety-nine percent uptime, sometimes called "two-nines", will earn 1 point, " five-nines" (99.999%) will earn 2 points, "Six-nines" (99.9999%) will earn 3 points. The node whose uptime is below

99% will earn 0 point. If a node's uptime changes, the node will broadcast a transaction to inform others about its uptime or availability status.

6. Offense score. This score is cumulative and not subject to time decay. Any nodes that made any serious attacks on other nodes or system, such as relaying the same transactions, will receive negative 3 points for its offense, at the same time, all its positive reputation scores will become 0, including its scores of leading, voting, transaction, availability, service, and resource. Defense score will remain. Its total reputation score will be the sum of defense score and offense score.

The total reputation score of a node will be the sum of its six scores obtained in individual attributes:

$$Reputation = Resource + Defense + Availability + Offense + Service + Function \tag{1}$$

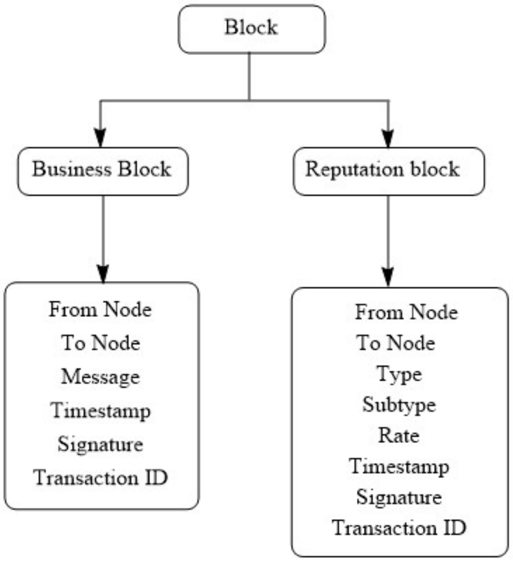

Figure 2: Block Structure

3.3 Consensus Mechanism

Figure 3 is the consensus flow chart.

1. Each node selects nodes with at least 90% of the highest total reputation score among all nodes as potential leaders.

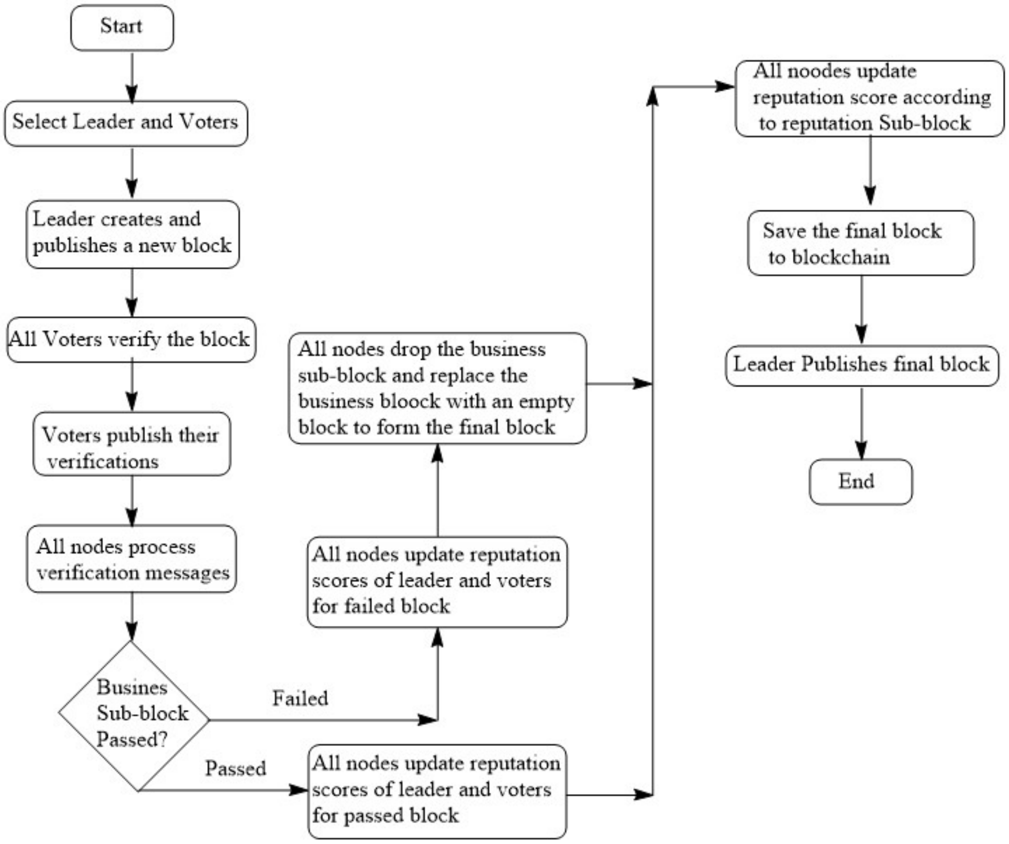
Start
Select Leader and Voters
Leader creates and publishes a new block
All Voters verify the block
Voters publish their verifications
All nodes process verification messages
Busines Sub-block Passed?
Failed
Passed
All nodes update reputation scores of leader and voters for failed block
All nodes drop the business sub-block and replace the business bloock with an empty block to form the final block
All nodes update reputation scores of leader and voters for passed block
All noodes update reputation score according to reputation Sub-block
Save the final block to blockchain
Leader Publishes final block
End

Figure 3: The Consensus Process

2. Select the node with the highest block creation score from the potential leaders as the leader.

3. Each node selects nodes with at least 10% of the highest total reputation score as potential voters.

4. From the potential voter pool, select nodes with at least 10% of the highest voting score as voters.

5. The leader creates a new block and publishes it, including voters.

6. The voters receive the block, verify the normal transactions, the reputation transactions, and publish the verification results

7. When consensus time expires, late responses will be ignored.

8. All the nodes process the received verification results, record the nodes

that have validated or failed the block or failed to send out their verifications.

9. If the block is validated, all nodes that have voted for validation will receive a voting score. Those voters that have failed the block or failed to send out their verification responses will receive negative voting points.

10. If the block is validated, all nodes will update any node's reputation according to the reputation transactions in the reputation sub-block.

11. The validated block will be added to the blockchain, including business and reputation sub-blocks, voters, and the maximum transactions in the blockchain so far.

12. If the block failed to be validated, all nodes that have voted for validation or failed to send out their verification responses will receive a negative voting score. Those voters that have voted to fail the block will receive the voting points. The failed block will be dropped.

13. The leader publishes the validated block.

4 Implementation and Evaluation

4.1 Block Composition and Block Voting

Each block contains two sub-blocks as shown in Figure 2. One is normal transaction block, named business block, and the other is reputation transaction block, named reputation block, which contains reputation transactions.

1. Business transaction structure contains information of from node, to node, message, timestamp, signature, and transaction ID, which is the hash of all the components in the transaction except the transaction ID itself.

2. Reputation transaction structure contains information of from node, to node, evaluation type, subtype, rate, timestamp, signature, and transaction ID, which is the hash of all the components in the reputation transaction except transaction ID itself. The evaluation type will be resource, defense, service, offense, and availability. If the evaluation type is resource, it will have three subtypes: CPU, memory, and storage. If the evaluation type is service, more detailed information about this service can be put into the subtype field.

3. A node will validate a block only when all business and reputation transactions are the same as its local business and reputation transactions.

4. A node's voting response contains voter, verification status, signature and timestamp. The verification status has two possible values: 0 and 1, with 0 meaning fail, and 1 denoting validation.

5. A block will be validated only when both sub-blocks are validated by at least 2/3 of the voters.

4.2 Consensus Algorithms

Table 1 lists the nomenclature for the consensus algorithms.

Algorithm 1 is run by all nodes to select the block leader and the group of voters.

Algorithm 1 Select leader and voters

```
Find the potential leaders and potential voters:
for all all nodes do
  if $NodeTR_i \geq 90\%$ of MaxTR then
    $Node_i \Rightarrow PLeaders$
  else if $NodeTR_i \geq 10\%$ of MaxTR then
    $Node_i \Rightarrow PVoters$
  end if
end for

Find the voters:
for all PVoters do
  if $NodeVR_i \geq 10\%$ of MaxVR then
    $Node_i \Rightarrow voters$
  end if
end for

Find the leader:
for all PLeaders do
  if $Node_i$ has MaxLR then
    $Node_i$ is the leader
  end if
end for
The block leader packs all its transactions in a defined period to create a new
block and publishes it
```

Algorithm 2 updates the block leader's and voters' reputation scores after a block is validated.

Algorithm 3 updates all nodes' transaction creation scores after a block is validated.

Algorithm 4 updates the block leader's and voters' reputation scores when a block fails to validate.

Algorithm 5 will run after a block is validated and will update all nodes' reputation according to the reputation transactions in the reputation block. The transactions in the reputation block are sorted by timestamps before the update to ensure the correct chronological order of transaction creation.

Table 1: Nomenclature

Variable	Meaning
Leader	Block leader
MaxNumBT	The highest number of transactions
MaxTR	Maximum total reputation score among all nodes
MaxTrans	The maximum transactions created by any node in a block
MaxVR	Maximum voting reputation score
NodeAR	Node's availability score
NodeCPU	Node's CPU core
NodeDFR	Node's defense reputation score
NodeMEM	Node's memory score
NodeRR	Node's resource reputation score
NodeSR	Node's service reputation score
NodeSTOR	Node's storage score
NodeTCR	Node's transaction creation score
NodeTR	Node's total reputation score
NodeTrans	The number of transactions created
NodeVR	Node's voting score
NumBT	Number of transactions in a block, including transaction in business and reputation blocks
perTrans	The percentage of transactions created by a node relative to MaxTrans.
PLeaders	Potential block leaders
PVoter	Potential voters
Rate	The value given to a reputation transaction
Voters	The group of voters
ir	Node r is evaluated in transaction i, *e.g.*, $NodeRR_{ir}$ represents one of node r's subtype resources being evaluated in transaction i; $NodeTR_{ir}$ - the total reputation score of node r being evaluated in transaction i; $NodeSR_{ir}$ - the service score of node r being evaluated in transaction i; $NodeDFR_{ir}$ - the defense score of node r being evaluated in transaction i

Algorithm 2 Update leader's and voters' reputation scores after a block's validation.

```
Find the voting point:
if $NumBT \geq 90\%$ of MaxNumBT then
    $point \leftarrow 3$
else if $NumBT \geq 50\%$ of MaxNumBT then
    $point \leftarrow 2$
else if $NumBT \geq 1$ then
    $point \leftarrow 1$
else
    $point \leftarrow 0$
end if

Update the leader's and all voters' voting and total reputation scores:
for all nodes in voters do
    if $Node_i$ is the block leader then
        $NodeVR_i \leftarrow NodeVR_i + point$
        $NodeLR_i \leftarrow NodeLR_i + point$
        $NodeTR_i \leftarrow NodeTR_i + 2\times\text{point}$
    else if $Node_i$ validated the block then
        $NodeVR_i \leftarrow NodeVR_i + point$
        $NodeTR_i \leftarrow NodeTR_i + point$
    else if $Node_i$ failed the block then
        $NodeVR_i \leftarrow NodeVR_i - point$
        $NodeTR_i \leftarrow NodeTR_i - point$
    else if $Node_i$ failed to send verification response then
        $NodeVR_i \leftarrow NodeVR_i - point$
        $NodeTR_i \leftarrow NodeTR_i - point$
    end if
end for
```

Algorithm 3 Update all node's transaction creation scores

```
Find MaxTrans created by a node in the block
for all Nodes in the network do
    $PerTrans \leftarrow (NodeTrans_i/MaxTrans) * 100$
    if $PerTrans \geq 80$ then
        $NodeTCR_i \leftarrow NodeTCR_i + 3$
        $NodeTR_i \leftarrow NodeTR_i + 3$
    else if $PerTrans \geq 50$ then
        $NodeTCR_i \leftarrow NodeTCR_i + 2$
        $NodeTR_i \leftarrow NodeTR_i + 2$
    else if $PerTrans \geq 1$ then
        $NodeTCR_i \leftarrow NodeTCR_i + 1$
        $NodeTR_i \leftarrow NodeTR_i + 1$
    end if
end for
```

Algorithm 4 Update reputation scores after a block fails to validate

```
Find the voting point:
if $NumBT \geq 90\%$ of MaxNumBT then
    $point \leftarrow 3$
else if $NumBT \geq 50\%$ of MaxNumBT then
    $point \leftarrow 2$
else if $NumBT \geq 1$ then
    $point \leftarrow 1$
else
    $point \leftarrow 0$
end if
Update all the voters' voting and total scores:
for all Node in voters do
    if $Node_i$ is the block leader then
        $NodeVR_i \leftarrow NodeVR_i - point$
        $NodeLR_i \leftarrow NodeLR_i - point$
        $NodeTR_i \leftarrow NodeTR_i - 2 * point$
    else if $Node_i$ failed the block then
        $NodeVR_i \leftarrow NodeVR_i + point$
        $NodeTR_i \leftarrow NodeTR_i + point$
    else if $Node_i$ validated the block then
        $NodeVR_i \leftarrow NodeVR_i - point$
        $NodeTR_i \leftarrow NodeTR_i - point$
    else if $Node_i$ failed to send verification response then
        $NodeVR_i \leftarrow NodeVR_i - point$
        $NodeTR_i \leftarrow NodeTR_i - point$
    end if
end for
```

Algorithm 5 Update reputation scores according to reputation transactions

```
for all Transactions in the reputation-block do
  if Resource Transaction then
    if Subtype Is CPU then
      $NodeCPU_{ir} = Rate$
    else if Subtype Is MEM then
      $NodeMEM_{ir} = Rate$
    else if Subtype Is STOR then
      $NodeSTOR_{ir} = Rate$
    end if
    $NodeTR_{ir} \leftarrow NodeTR_{ir} - NodeRR_{ir}$
    $NodeRR_{ir} = NodeCPU_{ir} + NodeMEM_{ir} + NodeSTOR_{ir}$
    $NodeTR_{ir} = NodeTR_{ir} + NodeRR_{ir}$
  else if Service Transaction then
    $NodeSR_{ir} \leftarrow NodeSR_{ir} + Rate$
    $NodeTR_{ir} \leftarrow NodeTR_{ir} + Rate$
  else if Available Transaction then
    $NodeTR_{ir} \leftarrow NodeTR_{ir} - NodeAR_{ir}$
    $NodeAR_{ir} \leftarrow$ Rate
    $NodeTR_{ir} \leftarrow NodeTR_{ir} + Rate$
  else if Defense Transaction then
    $NodeDFR_{ir} \leftarrow NodeDFR_{ir} + Rate$
    $NodeTR_{ir} \leftarrow NodeTR_{ir} + Rate$
  else if Offense Transaction then
    $NodeLR \leftarrow 0$
    $NodeVR \leftarrow 0$
    $NodeTCR_{ir} \leftarrow 0$
    $NodeRR_{ir} \leftarrow 0$
    Keep $NodeDFR_{ir}$ unchanged
    $NodeAR_{ir} \leftarrow 0$
    $NodeSR_{ir} \leftarrow 0$
    $NodeOFR_{ir} \leftarrow NodeOFR_{ir} - 3$
    $NodeCPUR_{ir} \leftarrow 0$
    $NodeMEMR_{ir} \leftarrow 0$
    $NodeSTOR_{ir} \leftarrow 0$
    $NodeTR_{ir} \leftarrow \leftarrow NodeDFR_{ir} + NodeOFR_{ir}$
  end if
end for
```

4.3 Experimental Environment

- The design was implemented on a MacBook Pro with MacOS Monterey, 2.7GHz Dual-Core Intel Core i5 processor, and 8GB 1867 MHZ DDR3 memory.
- The programming language was Golang, Version Go1.17.6 Darwin/AMD64.
- We created ten potential nodes to participate the implementation, eight of which were selected to vote, and the leader was included.
- Each node has its own directory. All the related information, *e.g.*, blockchain, node name, starting balance, private key, was saved in that location.
- Two nodes actively created transactions and performed consensus procedure.
- The nodes were connected to each other through their private keys.
- When a new node joined the network, it would receive an initial base balance - the initial scores, node name, and a private key.
- In our experiment, we limited our block size to 1.1 MB[23], which contains 2000 transactions.
- Both business and reputation transactions were created in the same loop. In each loop, at most only one transaction was created for each type.

4.4 Consensus Performance Evaluation

We evaluated the performance of R360 relative to other reputation systems in three scenarios. The first one is business transactions only, with no reputation transactions, which is named Function. A node's trustworthiness is measured by its leading score and voting score. The second scenario included also service reputation transactions. In this model, each run contains half business transactions and half service reputation transactions. The third is R360, which measures a node's trustworthiness from six dimensions, including function, resource, availability, defense, offense and service. In R360, half of the total transactions are business transactions, and half are reputation transactions.

As we can see from Figure 4, the time needed to do consensus increases as the number of transactions goes up. This is understandable because the complexities of our design are $O(N^2)$, O(NT), and (T^2), where N is the number of nodes in the network system, and T is the number of transactions in a block.

Table 2, 3, and 4 display Function, Service, and R360 performances, respectively. Table 5 lists the sub-reputation systems and their corresponding transaction typestransactions.

With the same number of transactions, R360 took the shortest consensus time, followed by service reputation, and function reputation model took the longest. This is because the complexity for the consensus algorithm is O(T^2).

Table 2: Function Performance

Trial	Number of Transactions	Consensus Time (Seconds)	Block Size	Block Status
1	4	6	3 KB	validated
2	50	6	29 KB	validated
3	100	6	56 KB	validated
4	200	6	111 KB	validated
5	400	7	222 KB	validated
6	600	9	333 KB	validated
7	800	12	443 kb	validated
8	1000	19	554 KB	validated
9	2000	52	1.1 MB	validated

Table 3: Service Performance

Trial	Number of Transactions	Consensus Time (Seconds)	Block Size	Block Status
1	4	6	3 KB	validated
2	52	6	30 KB	validated
3	100	6	58 KB	validated
4	200	6	114 KB	validated
5	400	6	228 KB	validated
6	800	9	454 kb	validated
7	1200	13	681 KB	validated
8	1600	19	908 KB	validated
9	2000	29	1.1 MB	validated

Table 4: R360 Performance

Trial	Number of Transactions	Consensus Time (Seconds)	Block Size	Block Status
1	4	6	3 KB	validated
2	52	6	30 KB	validated
3	100	6	57 KB	validated
4	200	6	113 KB	validated
5	400	7	226 KB	validated
6	800	8	451 kb	validated
7	1200	14	675 KB	validated
8	1600	18	901 KB	validated
9	2000	22	1.1 MB	validated

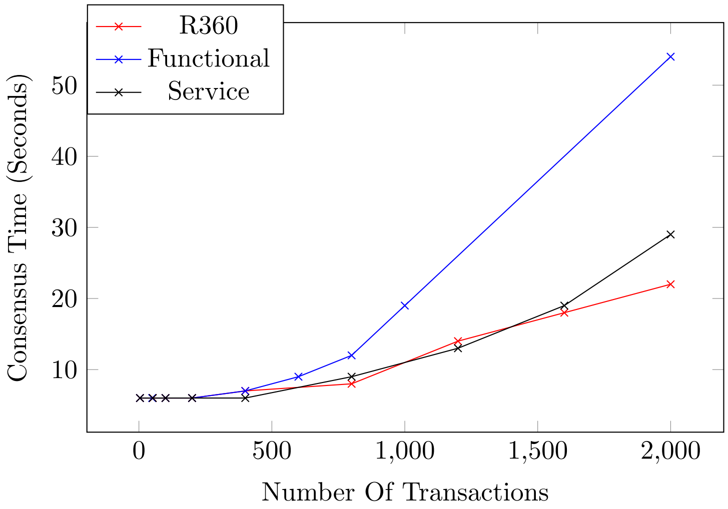

Figure 4: Consensus Performance

For R360, the total transactions are divided equally between business transactions and reputation transactions, and its complexity is half of $O(T^2)$. The service reputation transactions contain more information in its subtype than other reputation transactions, which is why service model took a little more time than R360. However, the time difference is insignificant. As the number of transactions goes up, the difference in time among the three models increases. R360 increases more slowly than the other two models. If R360 system can greatly increase a node's defensibility and reduce its offense ability, the reputation transactions can be reduced to contain only resource, availability, and service transactions, the block can have more business transactions while maintaining the security of the network. However, because R360 collects more information to measure a node's trustworthiness, more bandwidth will be required. More extensive experiments are needed to evaluate how R360 affects the overall network traffic.

4.5 Security Testing

We also carried out a security test with selfish attack in four scenarios: the leading node added an extra transaction to the to-be-proposed block, removed a transaction from the block, changed a transaction's rate value, and changed the evaluation's receiver. From Table 6, we can see that the consensus failed when the block leader modified a transaction. This is because the voters uncovered their local blocks were not the same as the altered blocks, so they failed the proposed blocks.

Table 5: Reputation System Transactions

No.	Sub-reputation System	Having Business Tranactions	Types of Reputation Transactions	Note
1	Function	Yes	0	Business transactions only
2	Resource	Yes	3	Resource Transaction: CPU, Memory, and Storage
3	Availability	Yes	1	Online time rate per year
4	Defense	Yes	1	Security breach transactions
5	Offense	Yes	1	Transactions attacking other nodes or network
6	Services	Yes	1	Service transactions to other nodes or system
7	R360	Yes	7	All the reputation transactions listed above

Table 6: Selfish Attack

Case	Original Transaction	Changes	Number of Transaction in Block	Block Status
1	Non Existing	Add a Transaction	40	Failed
2	Existing	Change Rate Value	40	Failed
3	Existing	Remove a Transaction	40	Failed
4	Existing	Change Receiver	40	Failed

5 Security Analysis

We tested selfish attack with R360 system. In this section, we analyze the threats of other common security attacks on R360.

- Bad-mouthing attack [15], [16]. A bad-mouthing attack provides dishonest recommendations to defame good nodes, which is the most straightforward attack [24]. In our protocol, a reputation transaction must obtain the consensus of 2/3 of voters in order to be effective, so malicious nodes cannot continuously malign a specific node or any other nodes and hope to improve its own trustworthiness ranking in a block in return.

- Replay attack. Replay attack attempts to reuse transactions and replay them in order to increase the impact of the same transaction. By carrying out such attack, a malicious participant can claim it has been involved in a transaction that is profitable for itself multiple times. It can also be used to undermine hostile participants. If the same transaction was used multiple times, other nodes that were involved in this transaction would take notice. If such malicious behavior is uncovered, the offensive node's positive reputation scores will be all wiped out.

- On-off attack. On-off attack refers to irregular behaviors of attackers, which means that malicious nodes can behave well or badly alternately in order to remain undetected while causing damage. A node's malicious behavior will result in a bad reputation score, which will negatively impact its total reputation score, which will subsequently reduce this node's chance to become a leader or a voter. This will negatively affect their reputation, reducing their chances of improving their reputation in the future.

- Sybil attack or newcomer attack. This kind of is harmful to almost all p2p networks. Attackers “legally” create multiple IDs. If one ID receives bad reputation because of offensive behaviors, it will switch to a new ID and start over. In R360, each node has a public key that is tied to a personal identification when a node is registered. Therefore, Sybil attack cannot occur.

- Flash attack. In flash attack, an attacker is able to obtain a temporary majority of computing power by renting enough mining capacity. This would break the security assumption of classic PoW-based systems. Our reputation system, however, is resilient to flash attacks. Even an attacker with high computing power, depending on when that attacker joins, needs to accumulate good scores over a very long period of time before being able to gain enough reputation to harm the system.

- Blockchain consistency and system liveness. Distributed systems, such as blockchains, have a concept of correctness, which includes two parts: safety and liveness. Liveness means that something good will happen.

In a blockchain consensus mechanism, liveness is the guarantee that all validators will agree on a value eventually. Safety is the guarantee that nothing bad will ever happen in the system. In terms of consensus, this means that no two processes/validators/actors will ever come up with different values. R360 is built on blockchain network concept, the voters must either validate or fail a block.

- Double spending attacks. Double-spending is the risk that a digital currency can be spent twice. It is a potential problem unique to digital currencies because digital information can be reproduced easily by savvy individuals who understand the blockchain network and the computing power necessary to manipulate it. Since digital currency is not used in R360 system, double spending attacks will not occur in our system.

- Eclipse attacks and isolated leaders. In eclipse attack, an attacker capable of delaying information that a victim expects to receive can launch double spending attacks and selfish attacks. Since all voters broadcast their results to the entire network, the leader will be able to receive the messages too. If a voter refuses to send its response, it will be punished with a negative reputation score.

6 Conclusion

A new system, R360, has been designed to thoroughly measure and maintain the reputation of nodes in an untrusted network. The design has been implemented to test the consensus performance and security. The results show that R360 takes less consensus time and enhances system security. Our experiments showed R360 can prevent selfish attack. Further analysis illustrates that it can prevent common attacks on blockchain reputation system. By putting on strict reputation evaluation rules, R360 can greatly enhance the security in untrusted blockchain smart grid. However, it also increases the network traffic. Our design was simulated on a single computer. Future work should implement R360 on real network systems to test its network performance and security. Additionally, the scoring system may need to be optimized. Further study on how such reputation scores affect individual node's behavior is also desirable.

References

[1] Zhaoyang Dong, Fengji Luo, Gaoqi Liang. Blockchain: a secure, decentralized, trusted cyber infrastructure solution for future energy systems. *Journal of Modern Power Systems and Clean Energy*, 6:958—-967, Jul. 2018.

[2] Cong T. Nguyen, Dinh Thai Hoang, Diep N. Nguyen, Dusit Niyato, Huynh Tuong Nguyen, Eryk Dutkiewicz. Proof-of-stake consensus mechanisms for

future blockchain networks: Fundamentals, applications and opportunities. *IEEE Access*, 7:85727–85745, 2019.

[3] N Anita., M Vijayalakshmi. Blockchain security attack: A brief survey. pages 1–6, 2019.

[4] Jiangshan Yu, David Kozhaya, Jeremie Decouchant, Paulo Esteves-Verissimo. Repucoin: Your reputation is your power. *IEEE Transactions on Computers*, 68(8):1225–1237, 2019.

[5] Yunhua He, Hong Li, Xiuzhen Cheng, Yan Liu, Chao Yang, Limin Sun. A blockchain based truthful incentive mechanism for distributed p2p applications. *IEEE Access*, 6:27324–27335, 2018.

[6] Blockchain consensus encyclopedia infographic. https://tokens-economy.gitbook.io/consensus/chain-based-proof-of-capacity-space/proof-of-reputation-por.

[7] Paul Resnick, Richard Zeckhauser. Trust among strangers in internet transactions: Empirical analysis of ebay' s reputation system. *The Economics of the Internet and E-commerce (Advances in Applied Microeconomics*, 11:127–157, 2002.

[8] Emanuele Bellini, Youssef Iraqi, Ernesto, Damiani. Blockchain-based distributed trust and reputation management systems: A survey. *IEEE Access*, 8:21127–21151, February 2020.

[9] Ferry Hendrikx, Kris Bubendorfer, Ryan Chard. Reputation systems: A survey and taxonomy. *Journal of Parallel and Distributed Computing*, 75:184–197, 2015.

[10] Muhammad Baqer Mollah, Jun Zhao, Dusit Niyato, Kwok-Yan Lam, Xin Zhang, Amer M.Y.M. Ghias, Leong Hai Koh, Lei Yang. Blockchain for future smart grid: A comprehensive survey. *IEEE Internet of Things Journal*, pages 1–1, 2020.

[11] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[12] Nick Szabo. Smart contracts : Building blocks for digital markets. 2018.

[13] Lanqin Sang, Henry Hexmoor. Information-centric blockchain technology for the smart grid. *International Journal of Network Security & Its Applications*, May 2021.

[14] Dodo Khan, Low Tang Jung, Manzoor Ahmed Hashmani. Proof-of-review: A review based consensus protocol for blockchain application. *International Journal of Advanced Computer Science and Applications(IJACSA)*, 12(3), 2021.

[15] Qianwei Zhuang, Yuan Liu, Lisi Chen, Zhengpeng Ai. Proof of reputation: A reputation-based consensus protocol for blockchain based systems. *IECC ' 19: 2019 International Electronics Communication Conference*, pages 131–138, July 2019.

[16] Fangyu. Gai, Baosheng. Wang, Wenping. Deng, Wei. Peng. Proof of reputation: A reputation-based consensus protocol for peer-to-peer network. *International Conference on Database Systems for Advanced Applications*, pages 666–681, 2018.

[17] Jie Duan, Mo-Yuen Chow. A resilient consensus-based distributed energy management algorithm against data integrity attacks. *IEEE Transactions on Smart Grid*, 10(5):4729–4740, 2019.

[18] Jose E. Fadul, Kenneth M. Hopkinson, Todd R. Andel, Christopher A. Sheffield. A trust-management toolkit for smart-grid protection systems. *IEEE Transactions on Power Delivery*, 29(4):1768–1779, 2014.

[19] Zheyuan Cheng, Mo-Yuen Chow. Reputation-based collaborative distributed energy management system framework for cyber-physical microgrids: Resilience against profit-driven attacks. pages 1–5, 2020.

[20] Tonghe Wang, Jian Guo, Songpu Ai, Junwei Cao. Rbt: A distributed reputation system for blockchain-based peer-to-peer energy trading with fairness consideration. *Applied Energy*, 295:117056, 2021.

[21] Wenjun Cai, Wei Jiang, Ke Xie, Yan Zhu, Yingli Liu, Tao Shen. Dynamic reputation–based consensus mechanism: Real-time transactions for energy blockchain. *International Journal of Distributed Sensor Networks*, 16(3):1550147720907335, 2020.

[22] Logan O.Mailloux, Michael R.Grimaila, John M.Colombi, Douglas D.Hodson, Gerald Baumgartner. Emerging trends in ict security. *ScienceDirect*, pages 5–23, 2014.

[23] Göbel, J. and Krzesinski, A.E. *Increased block size and Bitcoin blockchain dynamics.* 2017.

[24] Chrysanthos Dellarocas. Mechanisms for coping with unfair ratings and discrim- inatory behavior in online reputation reporting systems. *ICIS: International Conference on Computers and Information Systems*, pages 520–525, 2000.