

Received May 22, 2020, accepted June 3, 2020, date of publication June 8, 2020, date of current version June 30, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3000571

Defending Blockchain Forking Attack by Delaying MTC Confirmation

KAIYU WANG¹, YAN WANG², (Senior Member, IEEE), AND ZHENZHOU JI¹

¹School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China

²Department of Computing, Macquarie University, Sydney, NSW 2109, Australia

Corresponding author: Zhenzhou Ji (jizhenzhou@hit.edu.cn)

This work was supported by the National Scientific Research Project of China under Grant 2019603-001.

ABSTRACT In terms of security, blockchain technology faces the potential risks of enormous attacks, among which forking attack is the most common one. Because of the decentralized structure of a blockchain, the trust degree directly relies on the length of the blockchain, which makes the longest chain on the current network the most trusted chain (MTC). The forking attack is the kind of hazard behavior that aims to replace the MTC by launching an alternative chain to gain benefits. Blockchain plays a vital role in multiple applications, such as cryptocurrency, and hence such forking attack leads to huge financial loss if it succeeds. In this paper, we propose a novel MTC confirmation mechanism to defend the forking attack. First, we theoretically analyze the success probability distribution of the forking attack and propose an estimation model of the attack's expected profit. Then, we propose an arbitration mechanism to enforce branches from a fork to have competition. With the arbitration mechanism, we delay the confirmation of the MTC until the end of the competition. We set up an end threshold as the termination condition of the competition process to help the MTC become the winner. Finally, we define two dynamic difficulty factors and apply them for changing the difficulty of mining to reduce the value of the end threshold, which can improve the performance of the mechanism. The experiments demonstrate our proposed MTC confirmation mechanism can enhance the blockchain security by enforcing the expected profit of an attack to be negative. In the meantime, the mechanism balances well in both effectiveness and efficiency.

INDEX TERMS Blockchain, forking attack, MTC confirmation mechanism, dynamic difficulty factor.

I. INTRODUCTION

A. BACKGROUND

Blockchain is one of the emerging technologies in recent years; it not only triggers a new evolution of internet security, but also brings some new concepts of the decentralized network system to the public. For instance, the distributed ledger and smart contracts [12] are all the derivative applications of blockchain technique. Tracing back to 2008, Satoshi Nakamoto proposed a decentralized digital cryptocurrency system called Bitcoin [1]. The blockchain was proposed as an infrastructure of Bitcoin's transaction and circulation. However, this structure gains increasingly more popularities and attentions than Bitcoin itself. Since then, many researchers have started to explore the potential of blockchain, trying to apply it to many fields of applications, such

as communication, finance, military, and industrial control networks [13], [16].

Nakamoto's blockchain system maintains consensus in a different way from traditional *permissioned* distributed systems. The Bitcoin's core protocol "Nakamoto consensus" uses a *permissionless* setting, namely, every player, who is a participant of a blockchain, can join and leave the system without the permission of a centralized authority. This setting defines the spirit that a real decentralized peer-to-peer system should have. Every player can complete a transaction with any other player without the trusted third party (TTP) involved as a witness. However, it is necessary to provide methods to prevent one party of the transaction from breaching the contract. For instance, one seller may modify the transaction record afterwards and thereafter refuses to provide the purchased commodities or services.

A blockchain is essentially a data structure that stores public, immutable, and ordered ledger of records.

The associate editor coordinating the review of this manuscript and approving it for publication was Christian Esposito¹.

The *consensus protocol* is the method for blockchain to maintain such records [14]. According to the consensus protocol, players store transactions into a unit called *block*. A published block is linked to its preceding block to form a chain, which is called blockchain. In a block, the block head consists of the hash value of the head of its preceding block, a timestamp generated by the system, a difficulty target as target difficulty value of mining this block, a nonce working as a random value of mining and the hash value of the transaction data in the block body (called Merkle root). Hence, all blocks are connected as a chain. The structure of a blockchain is shown in Figure 1.

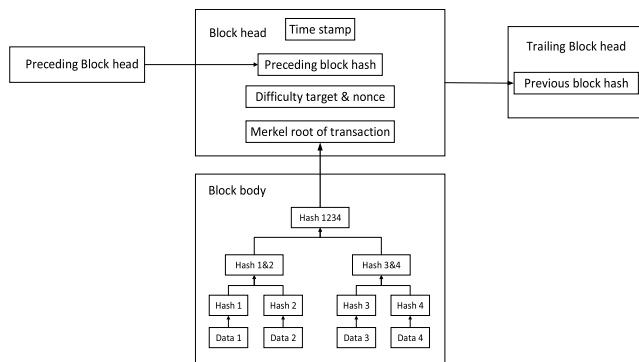


FIGURE 1. Blockchain structure.

Because of the irreversibility of the hash function [2], the transactions stored in a blockchain cannot be modified any more. However, the irreversible feature of the transaction records is not security enough. Most of the distributed systems rely on players' votes to deal with the affairs in which the majority party dominates the system. Because blockchain is a permissionless system, an attacker can become the majority party to manipulate the system by arbitrarily creating players. This kind of hazard behavior is called the *sybil attack* [28]. To protect blockchain against sybil attack, Nakamoto's system asks players to solve some public difficulty computing puzzles to generate blocks, the process of which is called *mining*. In the meanwhile, one block consumes a certain amount of computing power till confirmed its validity (the block needs to reach a specific depth on the chain, i.e. a specific number of trailing blocks), which is the trust source of the blockchain.

The computing power required for mining and confirmation is a good suppression to sybil attack. The chain which consumed the most computing power can be considered as the most trusted chain (MTC). If we assume the puzzles' difficulty fluctuates in a small range, the longest chain in the current network can be simply considered as the MTC [19].

The above consensus protocol relies on computing puzzles to mine and confirm blocks. Therefore, this protocol is called Proof of Work (PoW) [1]. However, PoW consumes huge electricity in the working process. To avoid the huge energy consumption in mining, Proof of Stake (PoS) [3] protocol is proposed. PoS protocol counts a player's coins and age of

coins as the guaranty instead of the computing power. In PoS, the MTC is the chain which contains the most guaranties, i.e. the longest chain. PoW and PoS are also the bases for many consensus protocols, such as Delegated Proof of Stake (DPoS) [31] and Proof of Activity (PoA) [5]. These derivative protocols have similar mechanism to their prototypes. Therefore, the longest chain can also be considered as the MTC in these protocols.

According to [19], the MTC should be the common parts for all players to store duplication in the local view, who follow the consensus protocol. That means a failure-free blockchain should have linear consistency [20]. Obviously, the MTC represents the authority in blockchain system instead of the centralized authority.

However, there are two main reasons limiting the development of blockchain, which are *intrinsic overhead* and *security hazards* [17], [18] respectively. Intrinsic overhead is caused by multiple synchronizations and verifications in the process of maintaining consensus. The more restrict of consensus the blockchain protocol requires, the more intrinsic overhead the system consumes. The protocol executes in rounds [14] or epochs [11] to achieve a periodical consensus, and the global consensus is maintained by achieving periodical consensus. However, the complex consistency maintenance process leads to enormous intrinsic overhead. Roughly speaking, a solo player with state-of-the-art equipment has to wait for an average of 2 years to get any rewards [26].

Due to the overlong payback period, more and more players give up the independent mining. They choose to contribute their computing power to form a mining pool. Each mining pool is managed by an agent, and all players share rewards and risks, which we call *joint mining*. The joint mining greatly destroys the decentralize feature of the blockchain and makes the blockchain system dominated by several mining pools. The attackers also use joint mining to attack the blockchain to increase the profit.

The security hazards caused by hostile attack are the other thorny problem of blockchain now. Many attackers try to attack the blockchain to gain benefits by utilizing the protocol design defects [7], [25]. However, because the MTC contains massive computing power or guaranties, the attacker cannot just modify the records by replacing an honest block on MTC. Therefore, launching an alternative chain to replace the MTC is one of a few effective attacking methods. With the occurrence of the alternative chain, the linear structure of blockchain becomes a fork structure. Hence, we call this kind of attack the *forking attack*.

Fork is a blockchain structure when the linear structure has been destroyed [20]. The common part of a fork is the *common prefix*, and the different parts are the *branches*. The longest branch in a fork is called *main-chain*, and the other branch is called *side-chain*. The MTC is constituted by the common part and the main-chain. The structure of fork is shown in Figure 2.

The forking attack is the kind of hazard behavior that aims to create a fork by launching a side-chain to replace the MTC.

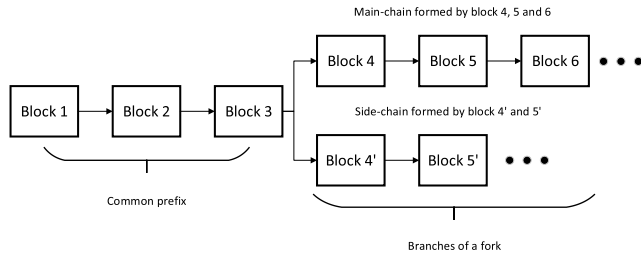


FIGURE 2. Fork structure in a blockchain.

The blockchain's authority will be changed if the attack succeeds. Then, the attacker can make dishonest records on the side-chain become valid to gain benefit.

Joint mining and *selfish mining* [25] are two methods the attacker used to complete the forking attack. Joint mining means the attacker forms his own mining pool to attack the blockchain. Selfish mining is the attacking method proposed by Eyal *et al.* [25]. In selfish mining, an attacker first mines a number of blocks and then selectively publishes some of them so as to dominate other branches, and thus gains benefits.

The forking attack has three fundamental features:

(i) *Simpleness*: In forking attack, the blocks mined by an attacker on the side-chain are also confirmed by computing power just like the blocks on the main-chain. The attacker just needs to mine blocks and selects a proper opportunity to publish them. Simpleness is the most obvious trait of a forking attack, which makes the attack launched without the need of any special hardware support. It means the barrier for the attacker is extremely low.

(ii) *Unpredictability*: The fork structure is caused by propagation delay, which is very common in a public blockchain due to the complex conditions of networks. The early stage of a forking attack can be hidden in enormous forks to gain a considerable advantage. The selfish mining increases the difficulty of detecting a forking attack in advance. That is the reason why the forking attack can hardly be predicted.

(iii) *High Success Probability*: If an attacker has enough computing capacity and a deliberate publishing plan, the attack will have a very high success probability. Actually, the forking attack does not contravene the consensus protocol. The current blockchain network has little means to protect main-chain from the threat of suddenly published side-chain.

The above three features are not the only reasons that makes the forking attack hardly be prevented. The egoism of non-stakeholder players makes the situation worse. The blockchain protocol tries to maintain consensus without centralized authority and TTP. The lacking of the authority and independent trusted party makes the MTC become the only authority. This means the attacker or any player who joins the attack can escape from punishment if the attack succeeds. This undoubtedly encourages the attacker to collude with non-stakeholder players to launch a forking attack.

In game theory, the model of players who only care about their personal interest is called Nash-equilibrium [27].

Reference [7] shows an attacker can increase the success probability by providing high incentives to attract non-stakeholder players by using this theory. However, most of the blockchain models considering the players' group are maintained in a relatively constant proportion, and thus they cannot give an accurate description of the forking attack.

In order to defend the forking attack, we consider the following three questions:

Q1: How can a forking attack succeed?

Q2: How to evaluate the profit of a forking attack?

Q3: How to convert the profit of a forking attack to be negative?

B. CONTRIBUTION

In this paper, we propose a novel MTC confirmation mechanism to defend the forking attack. First, we divide the whole attack into three phases and classify players into three groups according to their relations with the MTC.

Then, we define an attack profit estimation function based on the theory of Nash-equilibrium in which the players' behaviors are driven by the profit. This model can provide a more accurate description of forking attack.

Finally, we propose a novel MTC confirmation mechanism. In this mechanism, the confirmation of MTC will be delayed when encountering a fork structure. The main-chain will compete with the side-chain in the arbitration section until reaching the end threshold. The end threshold is set to promise the main-chain can win the competition if the fork structure is formed by a forking attack.

In addition, we define two new dynamic difficulty factors to increase the difficulty of consecutive mining on one branch, which significantly shortens the process of competition. With the help of dynamic difficulty factor, the competition process can be compressed to 25% of its original length. The forking attack can be defended by converting the expected profit into negative.

II. RELATED WORK

To defend the forking attack, there are three main methods which are: reducing fork, preventing joint mining and selfish mining. In this section, we review how the current protocols protect blockchain security through these methods respectively.

A. REDUCING FORK

To defend the forking attack, the most directly method is to reduce the generation of fork in blockchain system.

GeTrust [15] uses a guarantee-based trust model to reduce players' unconstrained behaviors which can be used to help blockchain maintain linear consensus and reduce forks. ByzCoin [34] uses joint signature to reduce the overhead of Practical Byzantine Fault Tolerance (PBFT) rounds to achieve a strong consensus to reduce fork production.

Researchers have proposed other methods to reduce the success probability of mining in a fork. Chou *et al.* [21] adopted a personalized difficulty adjustment to impede

players' consecutive mining on one branch to reduce the success probability of a forking attack. This method can indeed lower down the success probability, but cannot guarantee the honest players' security in the attack.

B. JOINT MINING

Joint mining is a notable feature of forking attack. Players in blockchain network are joint together to share rewards and risks. The joint mining of attackers can destroy the decentralized structure of blockchain. The attackers collude with non-stakeholders can further increase the success probability of the attack. The occurrence of joint mining is due to the overlong reward period and the expensive cost of mining blocks.

According to [34], blockchain network usually sets a propagation delay upper bound as the period of synchronization. The extension of the synchronization extends the reward period. Decker and Wattenhofer [33] also analyzed the impacts of propagation delay brought to blockchain system. According to the analysis, joint mining can be limited by reducing propagation delay.

Liu *et al.* [23] has proposed a probability verification called PvScheme. This study simplifies the block verification process to reduce impacts of the propagation delay, which greatly restrains the generation of blockchain forks. Bitcoin-NG [24] decouples blockchain operations into two planes: *leader election* and *transaction serialization*, to reduce the propagation delay. These two studies aim to shorten the miner's reward period. This hereafter decreases the miners' willing of joint mining, limiting the forking attack in some extent. However, all these methods have no strategy to punish the attacker who adopts joint mining. It has little methods to limit the malicious attackers' joint mining.

Decrease the mining cost is the other way to reduce the joint mining. Tendermint [30] is a blockchain system proposed by Jae Kwon which tries to maintain the consensus without mining. This system adopts Byzantine Fault Tolerance mechanism to generate blocks. However, Tendermint needs three rounds of votes to achieve Byzantine consensus. It saves the cost of mining, but requires a longer period to achieve consensus. PoS [3] and its derivative protocols also aim to provide consensus without consuming expensive cost in mining process. However, all these protocols are facing the threat of nothing at stake attack [32], in which the attackers can arbitrarily launch forks without the expense and punishment. This results in natural disadvantages in these protocols in defending the forking attack.

C. SELFISH MINING

Selfish mining is the other notable feature, the process of which can help an attacker to gain huge profit from launching a side-chain.

To defend this kind of attack, the fruitchain protocol [14] is proposed in which a fruit is a collection of transactions, and has a valid period before becoming corrupt (invalid). The transactions are stored in a fruit need to be linked to

a Nakamoto blockchain before the fruit becomes corrupt. Therefore, the attacker cannot keep his private fruitchain for too much time. However, it needs a more complex protocol to provide consensus for the fruitchain and the Nakamoto blockchain. And although the fruitchain protocol limits the length of private chain, the selfish mining can still increase forking attack's success probability in some extent.

III. PRELIMINARIES

In this section, we introduce our analysis of three issues. They are (a) the classification of players, (b) the phases of a forking attack, and (c) the probability distribution of attack phases.

A. CLASSIFICATION OF PLAYERS

Blockchain is a peer-to-peer network which consists of thousands to millions of players. This decentralized network relies on the majority of players following a consensus protocol to well maintain the blockchain's function. In ideal conditions, blockchain has a linear-consistency structure if all players mine and record honestly.

However, in practice, blockchain forks may be created because of hazard behaviors and environmental factors, such as propagation delay. The forks caused by environmental factors can be solved by the existing algorithms. For example, the RAFT algorithm proposed in [8] has been demonstrated to able to preserve consistency in a distributed system when considering internet delay.

The hazard behavior is the real potential threat due to its complexness in which a player acts the most crucial role. Therefore, before analyzing the forking attack, classifying players' behaviors and purposes is very important. Below we divide players into three groups: honest players, dishonest players and grey players.

1) HONEST PLAYERS

All the members of this group are honest and obey the consensus protocol to protect their benefits. They record the transactions truthfully and link their blocks to the MTC in the blockchain network.

2) DISHONEST PLAYERS

The dishonest players attack the blockchain by launching a side-chain. They are the attackers of the blockchain and also the sponsors of mining behaviors on side-chain. The dishonest players usually constituted by a group of players who aim to replace the valid honest records on the main-chain with invalid records on the side-chain to gain benefits. These players can decide whether and when to start and terminate such an attack.

3) GREY PLAYERS

Grey players obey consensus protocol in most of the time. They are usually non-stakeholders with the target. These players participate in transactions and aim to look for the biggest profit. Given the known cost of mining a block, the expected profit is the dominant factor in their behaviors.

They may join an attack when they are provided extra incentives that can make their benefit higher than those by honestly following the protocol.

B. PHASES OF ATTACK

Here we analyze the pattern of a forking attack. A forking attack is constituted of three phases: (1) preparing phase, (2) chasing phase, and (3) surpassing phase.

1) PREPARING PHASE

In this phase, the attacker launches a normal transaction T1 with huge value as the attack's target. T1 will be recorded into a main-chain block B1 by the miner of the block just like other normal transactions. Then B1 will be confirmed valid by reaching a specific depth α in the chain. After B1 has been linked on to the chain, the attacker mines a block B2 as the beginning block of his private chain recorded a transaction T2 launched by himself which is prepared to invalidate T1. In the meanwhile, the attacker keeps mining on his private chain to narrow the gap with the main-chain. The attacker chooses not to publish the whole private chain until B1 has been confirmed valid. Once B1 has been confirmed valid, T1 becomes formally effective. The attacker will hence receive his purchased commodity which is recorded on T1. Then the attacker publishes the whole private chain, which becomes the side-chain on the network. The preparing phase in a forking attack is shown in Figure 3.

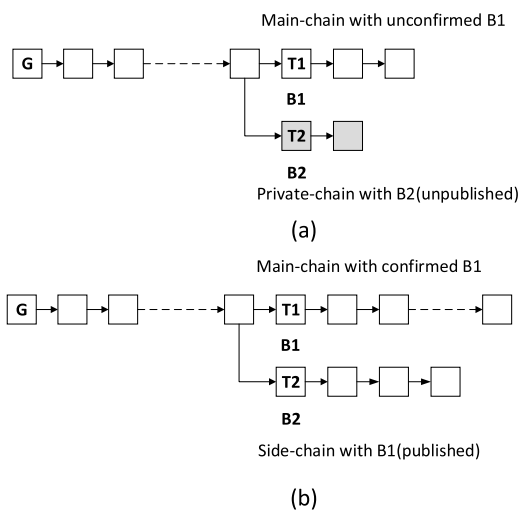


FIGURE 3. Preparing phase in a forking attack.

We assume the attacker mines β blocks in his private chain. Normally, $\beta < \alpha$, because the attacker's computing power is usually less than 50%. We name a variable $z = \alpha - \beta$ as the initial gap.

In rare cases, $\beta > \alpha$ which means the side-chain has surpassed the main-chain becoming the new main-chain. The forking attack completes without going into the rest two phases.

2) CHASING PHASE

Once the target block B1 is confirmed, the forking attack turns into the chasing phase. In this phase, the attacker mines on the side-chain and tries to attract grey players, by providing extra incentives from his profit. This phase terminates in two situations: (1) the side-chain equals the main-chain in length, or (2) the attacker's expected profit becomes negative.

3) SURPASSING PHASE

In this phase, if the side-chain surpasses the main-chain in length, and eventually becomes the longest branch on the fork. The original main-chain loses the status of the longest branch, and is no longer part of the MTC. All transactions including T1 recorded on the old main-chain become invalid when they are conflicted with the records on the new main-chain. The attacker can gain huge profit through replacing the transaction records with the side-chain surpassing the main-chain to make them invalid. The attack goes back to the chasing phase when the side-chain fails in surpassing the main-chain if expected profit is still positive.

C. PROBABILITY DISTRIBUTION OF ATTACK PHASES

The process of generating blockchain is essentially a random process. The detailed mid-process is full of randomness and indeterminacy. Therefore, using probability distribution to describe the development and operation from macroscopic aspect is a common method in the analyzing of blockchain. In the past, many researchers adopted a single Poisson distribution to describe the above all three phases in a forking attack. However, they do not consider the essential rule of Nash-equilibrium, which makes the computing power translated from main-chain to side-chain when the expected profit on side-chain is higher. In practice, the preparing phase can be seen as an approximate binomial distribution because the two branches (i.e., the main chain and the side chain) are non-interfering to each other. In the chasing phase, the probability distribution is a Continuous-Time Markov chain (CTMC) [16]. In the surpassing phase, the side-chain catches up with the main-chain. The whole forking attack can be seen as a mining race between main-chain and side-chain. The side-chain can win this race with an independent win in a discrete random event. The probability distribution becomes back to an independent binomial distribution.

The above analysis essentially answers Q1 of "How can a forking attack succeed?".

IV. ESTIMATION MODEL OF FORKING ATTACK

We devise an estimation model of the forking attack by defining three phases' probability distribution functions. We calculate the attack's success probability and evaluate how much extra incentive the attacker needs to provide to grey players.

A. PREPARING PHASE

The description about preparing phase in Section III is only for a simplified case. In practice, one attacker usually

launches many transactions which are recorded in several target blocks. Then the length of each of target blocks should be considered in the calculation and evaluation process. To simplify the analysis process, we here assume that the number of the target blocks is one. The attacker also needs to create many anonymous accounts for his trading behaviors to make it more reasonable to avoid the other players' suspicion. We simplify the specific process to consider all the anonymous buyers are working as a mining pool which is operated by an agent, namely, the attacker.

Once the target block has reached the special depth α , the attacker publishes his private chain as the side-chain. We assume the attacker could mine β blocks during the preparing phase. $z = \alpha - \beta$ is the initial gap; p and q are proportions of the computing power of the main-chain and side-chain respectively, which also represent the success probability of mining a block. Grey players can only choose to mine on the main-chain, because the attacker does not publish his side-chain on the network yet in this phase. Therefore, variable p is the sum of honest players' and grey players' computing power, and variable q is the computing power of the attacker. Obviously, $p = 1 - q$. The whole phase follows the binomial distribution. $P(q, \alpha, \beta)$ is the probability of side-chain surpasses the main-chain in the binomial distribution. The mining processes of the two branches are non-interfering to each other.

$$P(q, \alpha, \beta) = 1 \quad \text{if } \alpha = \beta = 0$$

$$P(q, \alpha, \beta) = \binom{\alpha + \beta - 1}{\beta} p^\alpha q^\beta \quad \text{otherwise}$$

B. CHASING PHASE

In the chasing phase, the attacker has published his private branch as a side-chain to challenge the status of the main-chain. Obviously, the success probability and the expected profit can hardly be very high due to the proportion of attacker's computing power. Therefore, the attacker needs to attract grey players to join the attack by sacrificing his own profit so as to provide extra incentive to grey players.

On the blockchain network, the rewards of mining constitute of two parts: (1) the reward of mining a block, and (2) the incentive of recording a transaction on the block [9]. On the main-chain, we normalized the reward and incentive of player's mining into 1 as the earning of mining. The attacker who wants to attract grey players needs to provide σ as the extra incentive of mining on the side-chain to make the expected profit of grey players higher than those of working honestly. In [7], Liao *et al.* have made a detailed analysis about the extra incentive function. However, his model has no grey players in the player classification.

According to the analysis in [10], the probability of z eventually reaching -1 is $\min(\frac{p}{q}, 1)^{z+1}$. It means the side-chain exceeds the main-chain in length. We denote it as p_z ; and the probability of the attack failing is $1 - p_z$. The computing powers for attacker, honest players and grey players are A , B_h and B_a respectively. $B_h + B_a = B$, $A + B_h + B_a = 1$.

The grey players' expected profit of mining on the main-chain is $\frac{(1-p_z)*B_a}{B} * 1$.

The expected profit of mining on the side-chain is $\frac{p_z*B_a}{A+B_a} * (1 + \sigma)$.

Then, we let $\frac{(1-p_z)*B_a}{B} * 1 \leq \frac{p_z*B_a}{A+B_a} * (1 + \sigma)$ to make the players' expected profit mining on the side-chain are higher than main-chain.

$$\sigma \geq \frac{(1-p_z)*B_a}{B} * \frac{A+B_a}{p_z*B_a} - 1$$

$$\geq \frac{\left(1 - \left(\frac{A}{B}\right)^{z+1}\right)*B_a}{B} * \frac{A+B_a}{\left(\frac{A+B_a}{B-B_a}\right)^{z+1}*B_a} - 1$$

The value of σ satisfying this polynomial is the lower bound of the extra incentive. We tabulate the lower bound of σ in different z with different computing power distributions in Table 1. The attacker's computing power A is shown in rows; and grey players' computing power B_a is in columns.

TABLE 1. Lower bound of extra incentive σ .

		Z=3					
		Computing power of B_a					
Computing power of A		0.05	0.1	0.15	0.2	0.25	0.3
	0.10	170.83	55.88	21.5	8.88	3.63	1.25
	0.15	59.18	22.80	9.45	3.89	1.38	0.18
	0.20	24.21	10.07	4.18	1.52	0.25	0
	0.25	10.71	4.48	1.67	0.32	0	0
	0.30	4.75	1.80	0.39	0	0	0
	0.35	1.85	0.42	0	0	0	0
		Z=2					
		Computing power of B_a					
Computing power of A		0.05	0.1	0.15	0.2	0.25	0.3
	0.10	29.29	13.2	6.49	3.23	1.49	0.5
	0.15	13.98	6.9	3.46	1.62	0.58	0
	0.20	7.31	3.69	1.76	0.66	0.01	0
	0.25	3.89	1.88	0.73	0.05	0	0
	0.30	1.95	0.78	0.08	0	0	0
	0.35	0.75	0.07	0	0	0	0
		Z=1					
		Computing power of B_a					
Computing power of A		0.05	0.1	0.15	0.2	0.25	0.3
	0.10	4.29	2.51	1.47	0.79	0.32	0
	0.15	2.65	1.56	0.86	0.38	0.03	0
	0.20	1.64	0.91	0.41	0.05	0	0
	0.25	0.94	0.43	0.07	0	0	0
	0.30	0.41	0.05	0	0	0	0
	0.35	0	0	0	0	0	0

It is obvious that the forking attack is easy to succeed if A and B_a are relatively high. The extra incentive is a reduction with the increase of attackers' and grey players' computing power. It also has a positive correlation with z .

The whole chasing phase is a CTMC with gap z as the state. z increases 1 with a probability $P_{up} = B - B_a$, which means the state of CTMC moving up a step. Similarly, z moves down a step with a probability $P_{down} = A + B_a$.

C_z is the probability of a side-chain reaching the status z . The mathematical expression of C_z is:

$$C_z = C_{z+1}P_{up} + C_{z-1}P_{down}$$

The whole process is presented in Figure 4.

$$C_z = \begin{cases} (P_{down}/P_{up})^z & \text{if } Z > 0, P_{down} \leq P_{up} \\ 1 & \text{if } Z > 0 \text{ or } P_{down} > P_{up} \end{cases}$$

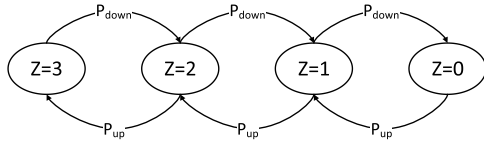


FIGURE 4. CTMD process in chasing phases.

The success probability of the chasing phase is:

$$C_z = \min(P_{down}/P_{up})^{\max(z,0)}$$

C. SURPASSING PHASE

When the attacker succeeds in the chasing phase; the two branches equal to each other in length. The probability distribution becomes back to the binomial distribution. If the side-chain exceeds the main-chain in length in this phase, the attack completes. The blockchain consensus protocol will then take the side-chain as the part of the MTC. The attacker can gain huge profit to make his target transaction invalid. If the side-chain fails in exceeding the main-chain in length, the attack goes back to the chasing phase with $z = 1$.

V. THE PROFIT OF FORKING ATTACK

In the above, we analyze the probability distribution of three phases in a forking attack. Now we can use the probability distribution to evaluate the expected profit of a forking attack.

In practice, a forking attack will launch more than one transaction as the target. The attacker will try to launch several transactions and make these transactions invalid in the same side-chain to maximize his profit. We estimate that every target transaction has a value v ; the amount of target transactions is n ; the whole value of all the targets is nv .

The expected profit consists of two parts: (1) the expected earning of a success attack, and (2) the expected cost of a failure attack. If the attack succeeds, all the block rewards of the side-chain will be valid; each block has a reward of 1. Also, the whole target's value nv should be added into the earnings. Otherwise, the attacker will lose all his blocks mined on the side-chain when the attack failed. Besides, the attacker has to provide incentives to grey players from his earning.

In the preparing phase, the number of blocks for a target turning valid is α . The expectation of blocks mined on side-chain in this phase is $\beta = \frac{q}{p} * \alpha$.

We assume that the side-chain needs to mine k blocks in the chasing phase to complete the attack. The evaluating function of k is based on four core ideas:

(1) the earning of single block for grey players to mine on the side-chain is $1 + \sigma$ in which 1 is the reward of mining a block and σ is the extra incentive from the attacker;

(2) the proportion of blocks mining in the chasing phase for the attacker and grey players is based on their computing power;

(3) the lower bound of k is z ;

(4) each block mined by grey players on side-chain causes σ net deficiency in the whole earning. The upper bound of k means the attacker sacrifices all his profit to complete the attack.

The whole earning of the attacker when the attack succeeds:

$$V_{earning} = (\beta + k + nv) - (1 + \sigma) * k * \frac{A}{A + B_a}$$

The cost of the attacker when the attack fails is:

$$V_{cost} = \beta + k * \frac{A}{A + B_a}.$$

The expected profit of the attacker for the whole attack:

$$V_{profit} = C_z * V_{earning} - (1 - C_z) * V_{cost}.$$

When $V_{profit} = 0$, we have the upper bound of k , $k_{up} = \frac{(2C_z - 1)\beta + C_z * nv}{\sigma * C_z * \frac{B_a}{A + B_a} - (2C_z - 1) * \frac{A}{A + B_a}}$, $k \in (z, k_{up})$.

The above analysis essentially answers Q2 of “How to evaluate the profit of a forking attack?”.

If we assume $nv = 50$, $\alpha = 5$, and $k = z$, the values of V_{profit} are listed in Table 2. The computing power of A is on the columns; the computing power of B_a is on the rows.

TABLE 2. $nv = 50$, $\alpha = 5$, $k = z$.

Computing power B_a	Computing power A				
	0.25	0.2	0.15	0.1	0.05
0.25	52.50	27.59	13.39	5.39	0.83
0.2	27.59	13.50	5.58	1.11	-1.45
0.15	8.28	2.02	-0.81	-2.17	-3.01
0.1	2.38	-0.35	-1.58	-2.22	-2.83
0.05	0.12	-0.96	-1.41	-1.70	-2.20

From Table 2, it is clear that the expected profit of a forking attack has a positive correlation with the computing power of A and B_a . In general, the expected profit is a function with two variables A and B_a . However, this correlation shows different changing trends when A and B_a working as the function variable respectively. The expected profit is a concave function of A when B_a is a constant. If A stays constant, the expected profit changes to a monotonic function of B_a .

The cause of the difference in the expected profit's changing trend is A and B_a has a different working range in an attack. A , as the computing power of the attacker, works in the whole attacking phases. When A has a higher value, the initial gap z is smaller. It also means a higher success probability of the attack and a lower extra incentive for grey players. The three impacts make the whole function have a more drastic reflection with the change of A . However, B_a only works in

the chasing phases. With every blocks mined by grey players, the expected profit has a net deficiency σ , which is the only relative part in the function. Therefore, the expected profit is a monotonic function of B_a .

VI. DELAYING MTC CONFIRMATION MECHANISM

On the blockchain network, all honest players consider the longest chain as the MTC. Some blockchain systems recognize the MTC as the only valid chain. Therefore, with the conclusion of Section V, we propose a novel MTC confirmation mechanism by delaying the MTC state confirmation to defend the forking attack.

On the original blockchain network, the valid confirmation of a new MTC completes when the main-chain status changes. This makes the main-chain hard to become a part of the MTC again if it has been surpassed. It brings the attacker some potential advantages due to the blockchain system losing the ability to self-correct. If we delay the confirmation of a new MTC, the attacker has to continue providing extra incentive, because of the extension of the chasing phase. This will greatly reduce the attacker's expected profit.

The answer to Q3: "How to force the profit of the forking attack negative?" is trying to reduce the attacker's expected profit by providing expensive extra incentives continually in the chasing phase. Then, we can make the profit of a forking attack negative, which consequently forces the sensible attacker to abandon the attack.

A. ARBITRATION MECHANISM

We propose an arbitration mechanism to delay the MTC confirmation. If side-chain surpasses the main-chain, the mechanism will not confirm the side-chain as the part of the MTC immediately.

The mechanism is activated when there is a fork in the blockchain. The early phase of a forking attack is always hidden in forks caused by other reasons, e.g. propagation delay. Therefore, we assume the blockchain is under a forking attack when there is a fork in the network. The mechanism is constituted by two parts: (a) arbitration section and (b) payment delaying mechanism. The arbitration section is a specialized section working as the execution area of the arbitration mechanism. In this section, the branches from the same fork will have a competition with each other when the arbitration mechanism is activated. The section records the whole competition process as the basis of determining effectiveness of validity of the winner and the invalidity of the loser. In the meanwhile, the records can be used as the criterion of simplify the competition process in case the attacker continues mining on a branch that has previously be deemed invalid. The payment delaying mechanism delays the payment of rewards of blocks mined in the arbitration section. The winner of the competition can receive the rewards according to the records of the competition and publish his blocks on the network as the part of the MTC when the competition terminates. The loser of the competition will have no rewards and the information of invalid blocks will

be broadcast to all the players in synchronization when the competition terminates.

In arbitration mechanism, the competition terminates in two situations:

- (1) One branch abandons mining behavior;
- (2) The competition process reaches the end threshold.

From the above analysis about the forking attack, we can see that if the attacker is the only player mining on the side-chain, the expected profit can hardly be positive. If the attacker wants to get help from grey players, he must provide extra incentive from his profit to make sure that the grey players can get higher expected profit from the attack than working honestly. The competition cannot continue endlessly due to such extra incentive. If the initial gap z is big enough, the extra incentive for the grey players is extremely expensive. The attacker has to consider to abandon the attack, because the attack's expected profit becomes valueless. Thus, the competition terminates in the first situation. The branch reaches the end threshold is the winner of the competition in situation 1.

The other way of competition termination is that the competition process reaches the end threshold. The end threshold is the length of the competition process that makes the attacker's expected profit negative. The difference between the source of computing power on the main-chain and the side-chain makes the mining on main-chain have a lower cost than side-chain, which guarantees that the main-chain can survive in the competition. Then, the branch which reaches the end threshold can be seen as the winner of the competition in situation 2, because the attacker has no reason to continue an attack when the expected profit becomes negative. Therefore, considering the two situations of competition termination, we can conclude that the branch that reaches the end threshold can be seen as the winner of the competition. Figure 5 shows the fundamental working process of arbitration mechanism.

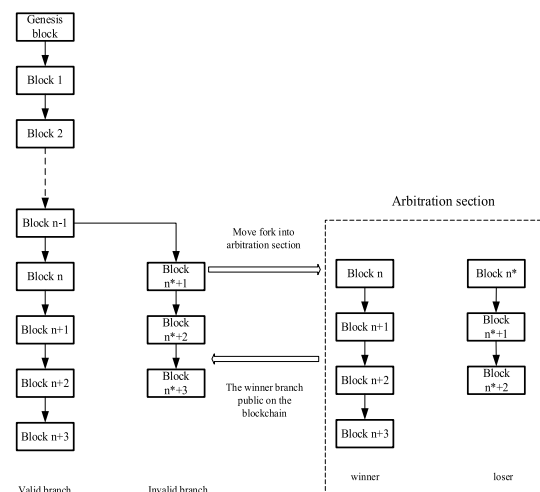


FIGURE 5. The arbitration mechanism.

The delaying of MTC confirmation means the extension of the chasing phase. It directly leads to the decrease of success probability and expected profit of the attack. Then the attacker will have to abandon mining on side-chain because of the negative expected profit.

B. FORK DETECTION

The pivotal factor of the arbitration mechanism is how to detect a fork on the blockchain. Kiayias *et al.* [11] propose a method to show the abstract structure of the blockchain network in a character string. Every block has a unique blockhead that can be used as the character of the string. Figure 6 shows the character string of a fork in the blockchain.

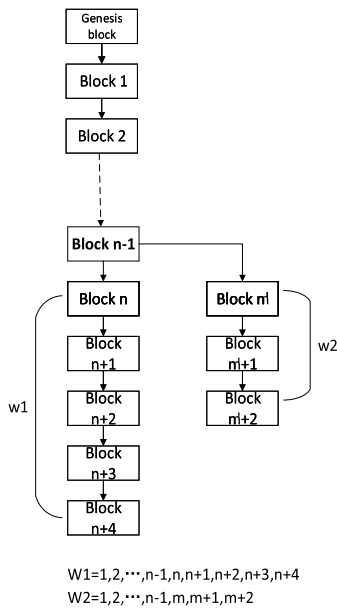


FIGURE 6. The character string of fork.

The character string is the abstraction of the blockchain structure. A non-forked blockchain has a linear structure which means one block should have only one trailing block. Therefore, the fork can be detected when there are two or more blocks with the same prefix. The relation of different branches' character string is very useful in analyzing the attack when there are several forks in the chain. If there are several competitions on arbitration sections, we can use the relation between these character strings of forks to simplify the competition. We propose two judgment conditions:

- (1) Each invalid branch makes its character string invalid;
- (2) If a branch's prefix has a non-void subset of an invalid string, then this branch can be determined as invalid.

C. END THRESHOLD

The end threshold is very important as the termination condition of the competition. The end threshold should be set to make sure the main-chain can survive from the competition. Then, we can use $\max(z, k_{up})$ as the end threshold of the competition. To calculate the end threshold, we need to estimate the target value nv , the value of extra incentive σ and the proportion of three group of players computing power.

Algorithm: arbitration

```

1: If fork detected
2: Arbitration ( $w1, w2$ )
3: While ( $!end\_threshold$ ) do
4:   Competition ( $w1, w2$ )// two branches have
     competition in arbitration sections
5:   use judgment condition to simplify the competition
6:   simplify ( $w1, w2$ )
7:   If the competition over
8:     Return winner
9:   loser string  $w \rightarrow invalid\_stack$ 
10:  Else if one branch gives up
11:    Return winner
12:  loser string  $w \rightarrow invalid\_stack$ 
13: If ( $end\_threshold$ )
14:  Return winner
15:  loser string  $w \rightarrow invalid\_stack$ 

```

FIGURE 7. The pseudo-code of the arbitration mechanism.

Algorithm: fork detecting

```

1: get character string ( $blockchain, timestamp$ )
2: Return  $w$ 
3: traversal the string  $w$ 
4: While ( $child(w[i]) \neq null$ ) do
5:    $child(w[i+1])$ 
6:   If  $child[w] > 1$  then
7:     Return branches strings
8:   Else  $child[w] = null$ 
9:   Return 0

```

FIGURE 8. The pseudo-code of fork detecting.

Algorithm: simplify

```

1: while ( $!end\_threshold$ ) do
2:   for each string  $w_i$ 
3:     If ( $w_i \parallel invalid\_stack$ )  $\neq 0$ 
4:        $w_i \rightarrow invalid\_stack$ 

```

FIGURE 9. The pseudo-code of simplifying the arbitration.

The target value nv is the total transactions' value of main-chain's transaction in which are conflicted with the transactions on side-chain. Normally, they are transactions involving huge amounts of money and large quantities of commodities. The value of extra incentive σ can be obtained from network communication, because the attacker needs to broadcast this information to the whole network to attract grey players. The attacker's computing power can be estimated by the initial gap z . The proportion of attacker's computing power approximately equals to $\frac{\beta}{\alpha+\beta}$. The proportion of honest player and grey players' computing power can be estimated according to the number of target stakeholder players and the information we have obtained. Then, we have the obtained all values of variables needed to calculate the end threshold.

In practice, many attacks may be abandoned before reaching the end threshold, because the attack has a low expected profit (non-negative). As we mentioned above, the whole attacking process is a random process that is the reason why we use the probability distribution to describe the attack. The expected profit is the mathematical expectation of multiple independent attacks' profit. For a single independent attack, the attacker may have a negative actual profit when the expected profit is still positive (but very low). With the more attacks launched, the actual profit will gradually converge to the expected profit.

Therefore, we use the expected profit to calculate the value of the end threshold $\max(z, k_{up})$ that makes this value a looser upper bound of the actual situation. We calculate the value of end threshold with different nv . The values of the end threshold in different situations are presented on Tables 3(a)-3(c). The computing power of A is on the columns; the computing power of B_a is on the rows.

TABLE 3. (a). $nv = 20, \alpha = 5$. (b) $nv = 30, \alpha = 5$. (c) $nv = 40, \alpha = 5$.

		Computing power A				
Computing power B_a		0.25	0.2	0.15	0.1	0.05
	0.25	54	12	5	5	5
	0.2	41	12	5	5	5
	0.15	7	4	5	5	5
	0.1	4	4	5	5	5
	0.05	4	4	5	5	5

		Computing power A				
Computing power B_a		0.25	0.2	0.15	0.1	0.05
	0.25	81	18	6	5	5
	0.2	62	18	6	5	5
	0.15	11	4	5	5	5
	0.1	4	4	5	5	5
	0.05	4	4	5	5	5

		Computing power A				
Computing power B_a		0.25	0.2	0.15	0.1	0.05
	0.25	107	25	8	5	5
	0.2	82	25	9	5	5
	0.15	15	4	5	5	5
	0.1	6	4	5	5	5
	0.05	4	4	5	5	5

The end threshold raises with the increase of nv , because the more valuable target can withstand the longer competition. nv raises the expected profit, also provides the attacker more property for the extra incentive. The higher nv means the attacker and grey players can mine on the side-chain more unhurried. Therefore, we need a higher end threshold to guarantee the main-chain can survive from the competition.

D. DYNAMIC DIFFICULTY FACTOR

From the above tables, we find that the end threshold can be extremely high in some situations. The overlong competition process will drag down the efficiency of the blockchain

and lengthen honest player's payback period. As we know, the above two problems are always the obstacle of the development of blockchain. Therefore, we propose a dynamic difficulty factor to shorten the competition process by reducing the probability of consecutively mining on one branch.

In blockchain system, the sign of a success mining is the player computes a hash value which satisfies the difficulty target with nonce as input. The dynamic difficulty factor working as a mining difficulty adjustment tool to dynamic adjust the mining difficulty on branches according to the block records in the blockchain system. If blocks are consecutively mined on one branch, the dynamic difficulty factor will adjust the mining difficulty on this branch to reduce the transition probability of z .

We define two dynamic difficulty factors $D1$ and $D2$. $D1 = (l, \theta)$ can be considered as a function of preceding block label l and the difficulty coefficient θ . The preceding block label is used to distinguish block mined on different branch. We here use M and S to represent this block mined on the main-chain and side-chain. If the preceding block's is mined on one branch, the dynamic difficulty factor will adjust the mining difficulty on this branch with coefficient θ . $D1$ only considers the preceding blocks label. For example, if the side-chain has mined one block, then the mining difficulty of next block for side-chain is θ time harder than original difficulty.

In $D2 = (l, n, \theta)$, the new variable n is the number of consecutive blocks on one branch. The mining difficulty on one branch has positive correlation with the number of preceding consecutive blocks. The mining difficulty of this branch becomes $n * \theta$. For example, the preceding two consecutive blocks on side-chain will lead to the side-chain's mining difficulty of next block is $2 * \theta$ times harder than original difficulty. Even if side-chain's computing power is extremely high, it is difficulty to shorten the gap z in a short time under $D2$. In the experiment, we set the difficulty coefficient $\theta = 2$. D_M is the difficulty factor of the main-chain; D_S is the factor of the side-chain.

The illustration of $D1(l, \theta)$ and $D2(l, n, \theta)$ is shown in Figure 10.

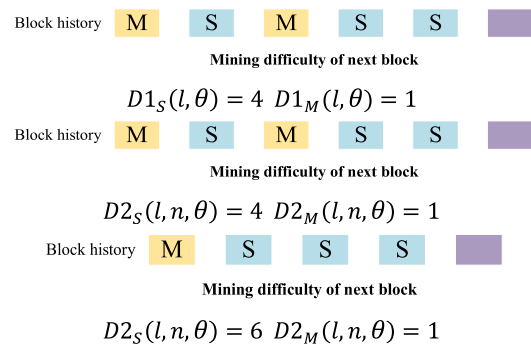


FIGURE 10. Dynamic difficulty factors $D1$ and $D2$.

With the dynamic difficulty factors, the status of CTMC in chasing phase tends to transfer between a locally small range. Two consecutive blocks on one branch can cause a stable state transfer, because the next block will have huge probability being mined by the other branch due to the difference of mining difficulty. Then, we have the new state transition probability of z in chasing phase: $P_{up'} = P_{up}/D(m)$ and $P_{down'} = P_{down}/D(s)$.

The new end thresholds after using dynamic difficulty factor $D1$ are shown in Table 4(a-c). The computing power of A is on the columns; the computing power of B_a is on the rows.

TABLE 4. (a) $nv = 20, \alpha = 5$. (b) $nv = 30, \alpha = 5$. (c) $nv = 40, \alpha = 5$.

		Computing power A				
Computing power B_a		0.25	0.2	0.15	0.1	0.05
	0.25	13	5	5	5	5
	0.2	14	4	5	5	5
	0.15	4	4	5	5	5
	0.1	4	4	5	5	5
	0.05	4	4	5	5	5

		Computing power A				
Computing power B_a		0.25	0.2	0.15	0.1	0.05
	0.25	20	6	5	5	5
	0.2	21	7	5	5	5
	0.15	6	4	5	5	5
	0.1	4	4	5	5	5
	0.05	4	4	5	5	5

		Computing power A				
Computing power B_a		0.25	0.2	0.15	0.1	0.05
	0.25	27	9	5	5	5
	0.2	29	10	5	5	5
	0.15	8	4	5	5	5
	0.1	4	4	5	5	5
	0.05	4	4	5	5	5

The new end thresholds using dynamic difficulty factor $D2$ are presented in Table 5(a-c). The computing power of A is on the columns; the computing power of B_a is on the rows.

From the above tables we can find that the end threshold when using the dynamic difficulty factors is significantly lower than without using the factors. The difficulty of consecutive mining on one branch makes attacker can hardly narrow the gap z , which reduces the attacker's expected profit and shortens the end threshold. The factor $D1$ and $D2$ can shorten the competition process to deal with different kinds of blockchain system.

The initial gap z enforces the attacker to have consecutive mining on side-chain to complete his attack. The alternative growth between main-chain and side-chain only keeps the z unchanged and consumes the profit of the attack. Therefore, we set a dynamic difficulty factor to prevent any consecutive mining. We propose two difficulty factors $D1$ and $D2$ for different kinds of blockchain systems. Some lightweight blockchain systems have shorter mining period and lower propagation delay. Thus, the same initial gap z cannot

TABLE 5. (a) $nv = 20, \alpha = 5$. (b) $nv = 30, \alpha = 5$. (c) $nv = 40, \alpha = 5$.

		Computing power A				
Computing power B_a		0.25	0.2	0.15	0.1	0.05
	0.25	4	5	5	5	5
	0.2	5	4	5	5	5
	0.15	4	4	5	5	5
	0.1	4	4	5	5	5
	0.05	4	4	5	5	5

		Computing power A				
Computing power B_a		0.25	0.2	0.15	0.1	0.05
	0.25	7	5	5	5	5
	0.2	8	4	5	5	5
	0.15	4	4	5	5	5
	0.1	4	4	5	5	5
	0.05	4	4	5	5	5

		Computing power A				
Computing power B_a		0.25	0.2	0.15	0.1	0.05
	0.25	10	5	5	5	5
	0.2	11	4	5	5	5
	0.15	4	4	5	5	5
	0.1	4	4	5	5	5
	0.05	4	4	5	5	5

provide the same level of advantage for main-chain as some long mining period and high propagation delay blockchain systems. Therefore, we need an aggressive difficulty factor $D2$ for lightweight blockchain systems. In the meanwhile, we propose a relatively mid difficulty factor $D1$ for high mining difficulty blockchain system, such as Bitcoin to prevent players consecutive mining with as little impact on system performance as possible.

Although the difficulty factor makes it difficult to have a consecutive mining on both main-chain and side-chain, the main-chain still has advantage because of the initial gap z . The difficulty factor drags down the transition probability of z in CTMC. The attacker will need more computing power to narrow the gap. In the meanwhile, the lower transition probability of z also means the lower success probability of the attack. It directly leads to the decrease of the attacker's extra earning. Therefore, the attacker can only support a shorter chasing phases for the same value of the target.

Therefore, the arbitration mechanism will compute the new end threshold with the new transition probability $P_{up'}$ and $P_{down'}$ to adjust length of competition process. It is obvious that the end threshold of the competition will have a smaller value than normal situation when the attacker has the same computing power and target value.

E. PERFORMANCE ANALYSIS

Nowadays, despite blockchain is getting mature, efficiency remains one of the main challenges of this technology. In particular, some protecting methods and consensus protocols often increase the performance burden. In this paper, we use the average transaction confirmation time to measure the performance. The normal blockchain handling capacity can

be defined as $\frac{\alpha * T}{n}$ where T is the time for the network to generate a block; n is the number of transactions included in one block; α is the confirmation depth; K is the end threshold of the competition. Then, the average confirmation time for a transaction when there is a forking attack is:

$$\frac{\frac{\alpha}{2} * (\alpha - 1) + (K - \alpha) * 2K}{K * n}.$$

In the experiment, we use Bitcoin's consensus mechanism as the benchmark, which $n = 10$, $\alpha = 5$, $T = 600s$. The average transaction confirmation delay is 300s. In the arbitration section, the last α blocks' confirmation time is 300s; the rest blocks' confirmation time is according to the length of competition process. The average transaction delaying time of our MTC confirmation mechanism in different situations is shown in Figure 11(a).

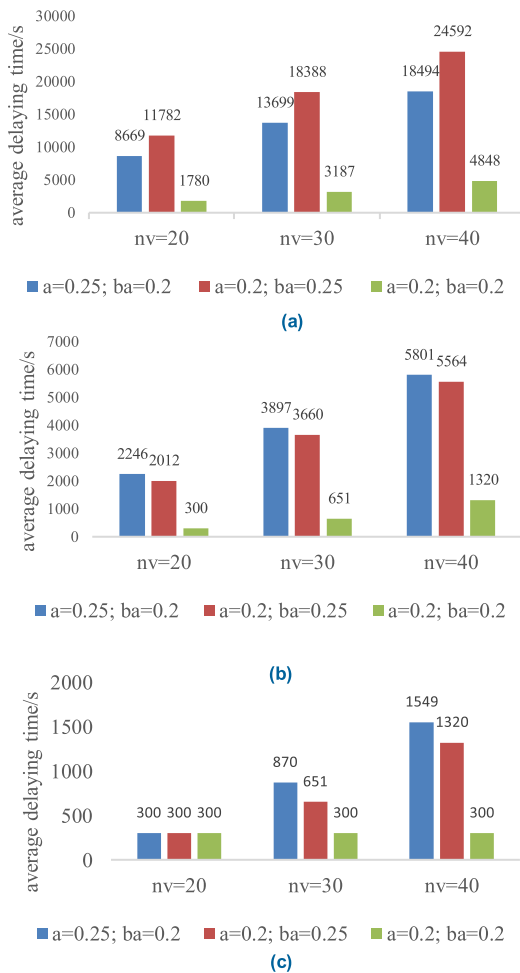


FIGURE 11. (a) Average transaction delaying time. (b) Average transaction delaying time of D1. (c) Average transaction delaying time of D2.

The average transaction delaying time of using dynamic difficulty factor $D1$ is shown in Figure 11(b). The average transaction delaying time of using dynamic difficulty factor $D2$ is shown in Figure 11(c).

From the above figures, we calculate the average transaction delaying time in a forking attack and the average

transaction delaying time when using dynamic difficulty $D1$ and $D2$. In Figure 11(a), the normal average delaying time for one transaction is really high when encountering a forking attack. The high delaying time is caused by our MTC confirmation mechanism which delays the validation of transaction. The mechanism sacrifices blockchain's performance to defend the forking attack. The drastic competition between main-chain and side-chain makes the competition process can be extremely long when the side-chain has relatively high computing power, which causes the extremely long average transaction delaying time.

From Figure 11(b-c), it is obviously that our dynamic difficulty factors $D1$ and $D2$ can significantly reduce the competition process by decreasing the transition probability of state z in chasing phase. $D1$ can shorten the competition process into 25% of its original length by prevent attacker's consecutive mining on the side-chain. $D2$ as a more restrict difficulty factor than $D1$ can even limits the length of competition process to a shorter one. The experiments demonstrate our dynamic difficulty factor can significantly reduce the average transaction delaying time by decreasing the length of the competition process, which has proven our mechanism's efficiency and effectiveness.

VII. CONCLUSION

The forking attack is the hazard behavior that the attacker creates an alternate chain i.e. side-chain to replace the state of MTC. The forking attack becomes the only few methods to attack the system, because blocks use hash function to link as a chain. We build a profit estimation model to evaluate the success probability and the expected profit of a forking attack. Then, we analyze the relationship between the probability C_z and the computing power of the dishonest player A and the grey player B_a . The mechanism we proposed can defend the forking attack by delaying the confirmation of MTC, which forces the attacker to provide more extra incentive to grey players to decline their profit. The whole model is based on theory of Nash-equilibrium, which means all the players on the network only consider their personal profit. The attacker will abandon the attack when the expected profit is negative. The grey players will choose to mine on the main-chain or side-chain by comparing which choice has a higher profit.

The delaying MTC confirmation mechanism forces the branches to have a competition before authenticate a new MTC. From the above analysis, we found that the end threshold can help main-chain to win the competition even when the side-chains' computing power is close to 50%. If a fork is detected, we assume that the blockchain is under a forking attack. The two branches of a fork will have a competition in the arbitration section until someone of them gives up or reaches the end threshold. In the arbitration procedure, we propose two judgment conditions to simplify the competition process. Then, the dynamic difficulty factor will adjust the difficulty of consecutive mining on branches to prevent side-chain narrow the gap z . With the mechanism we proposed, the forking attack can be defended by forcing its

profit into negative. Furthermore, the competition process in arbitration sections can be controlled within a limited size even when the computing power of two branches are close.

REFERENCES

- [1] S. Nakamoto, *Bitcoin: A Peer-to-Peer Electronic Cash System* (Consulted), vol. 1. 2012. [Online]. Available: <http://www.bitcoin.org/bitcoin.pdf>
- [2] N. T. Courtois, M. Grajek, and R. Naik, "Optimizing SHA256 in bitcoin mining," in *Proc. Int. Conf. Cryptogr. Secur. Syst.*, 2014, pp. 131–144, doi: [10.1007/978-3-662-44893-9_12](https://doi.org/10.1007/978-3-662-44893-9_12).
- [3] *Proof-of-Stake-FAQs*. [Online]. Available: <https://github.com/ethereum/wiki/wiki/Proof-of-Stake-FAQ>
- [4] M. Castro and B. Liskov, "Practical byzantine fault tolerance and proactive recovery," *ACM Trans. Comput. Syst. (TOCS)*, vol. 20, no. 4, pp. 398–461, Nov. 2002, doi: [10.1145/571637.571640](https://doi.org/10.1145/571637.571640).
- [5] I. Bentov, C. Lee, A. Mizrahi, and M. Rosenfeld, "Proof of activity: Extending Bitcoin's proof of work via proof of stake [extended abstract]," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 42, no. 3, pp. 34–37, Dec. 2014.
- [6] C. Pinzón and C. Rocha, "Double-spend attack models with time advantage for bitcoin," *Electron. Notes Theor. Comput. Sci.*, vol. 329, pp. 79–103, Dec. 2016.
- [7] K. Liao and J. Katz, "Incentivizing blockchain forks via whale transactions," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.*, Sliema, Malta, Apr. 2017, pp. 264–279.
- [8] D. Ongaro and J. Ousterhout, "In search of an understandable consensus algorithm," in *Proc. USENIX ATC USENIX Conf. USENIX Annu. Tech. Conf.*, Jun. 2014, pp. 305–320.
- [9] O. Ersoy, Z. Ren, Z. Erkin, and R. L. Lagendijk, "Information propagation on permissionless blockchains," *Trans. Propag. Permissionless Blockchains, Incentive Routing Mech.*, vol. abs/1712.07564, pp. 20–30, Jun. 2018, doi: [10.1109/CVCBT.2018.00008](https://doi.org/10.1109/CVCBT.2018.00008).
- [10] M. Rosenfeld, "Analysis of hashrate-based double spending," 2014, *arXiv:1402.2009*. [Online]. Available: <http://arxiv.org/abs/1402.2009>
- [11] A. Kiayias, A. Russell, B. David, and R. Oliynykov, "Ouroboros: A provably secure proof-of-stake blockchain protocol," in *Proc. Annu. Int. Cryptol. Conf.*, Santa Barbara, CA, USA, Aug. 2017, pp. 357–388.
- [12] J. Zou, Y. Wang, and M. A. Orgun, "A dispute arbitration protocol based on a peer-to-peer service contract management scheme," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, San Francisco, CA, USA, Jun. 2016, pp. 41–48.
- [13] X. Wang, X. Zha, W. Ni, R. P. Liu, Y. J. Guo, X. Niu, and K. Zheng, "Survey on blockchain for Internet of Things," *Comput. Commun.*, vol. 136, pp. 10–29, Feb. 2019, doi: [10.1016/j.comcom.2019.01.006](https://doi.org/10.1016/j.comcom.2019.01.006).
- [14] R. Pass and E. Shi, "FruitChains: A fair blockchain," in *Proc. ACM Symp. Princ. Distrib. Comput.*, Jul. 2017, pp. 315–324, doi: [10.1145/3087801.3087809](https://doi.org/10.1145/3087801.3087809).
- [15] X. Meng and D. Liu, "GeTrust: A guarantee-based trust model in chord-based P2P networks," *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 1, pp. 54–68, Jan. 2018.
- [16] K. Zheng, Y. Liu, C. Dai, Y. Duan, and X. Huang, "Model checking PBFT consensus mechanism in healthcare blockchain network," in *Proc. 9th Int. Conf. Inf. Technol. Med. Edu. (ITME)*, vol. 1, Oct. 2018, pp. 877–881.
- [17] M. Piekarska and H. Halpin, "Introduction to security and privacy on the blockchain," in *Proc. IEEE Eur. Symp. Secur. Privacy Workshops (EuroS PW)*, Paris, France, Apr. 2017, pp. 1–3.
- [18] A. Sapirshstein, Y. Sompolsky, and A. Zohar, "Optimal selfish mining strategies in bitcoin," in *Financial Cryptography and Data Security*. Berlin, Germany: Springer, Feb. 2016, pp. 515–532, doi: [10.1007/978-3-662-54970-4_30](https://doi.org/10.1007/978-3-662-54970-4_30).
- [19] J. A. Garay, A. Kiayias, and N. Leonardos, "The bitcoin backbone protocol: Analysis and applications," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.-Adv. Cryptol.*, Sofia, Bulgaria, 2015, pp. 281–310.
- [20] E. Blum, A. Kiayias, C. Moore, S. Quader, and A. Russell, "Linear consistency for Proof-of-Stake blockchains," 2019, *arXiv:1911.10187*. [Online]. Available: <http://arxiv.org/abs/1911.10187>
- [21] C.-N. Chou, Y.-J. Lin, R. Che, H.-Y. Chang, I.-P. Tu, and S. W. Liao, "Personalized difficulty adjustment for countering the double-spending attack in proof-of-work consensus protocols," in *Proc. IEEE Int. Conf. Internet Things (iThings) IEEE Green Comput. Commun. (GreenCom) IEEE Cyber, Phys. Social Comput. (CPSCom) IEEE Smart Data (SmartData)*, Jul./Aug. 2018, pp. 1456–1462, doi: [10.1109/Cybermatics.2018.00247](https://doi.org/10.1109/Cybermatics.2018.00247).
- [22] A. Back, M. Corallo, and L. Dashjr, "Enabling blockchain innovations with pegged sidechains," *Tech. Rep.*, 2014, p. 72. [Online]. Available: <http://www.opensciencereview.com/paper/123/enablingblockchain-innovations-with-pegged-sidechains>
- [23] B. Liu, Y. Qin, and X. Chu, "Reducing forks in the blockchain via probabilistic verification," in *Proc. IEEE 35th Int. Conf. Data Eng. Workshops (ICDEW)*, Apr. 2019, pp. 13–18.
- [24] I. Eyal, A. E. Gencer, E. G. Sirer, and R. Van Renesse, "Bitcoin-NG: A scalable blockchain protocol," in *Proc. 13th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, 2016, pp. 45–59.
- [25] I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.* Berlin, Germany: Springer, 2014, pp. 436–454.
- [26] (2017). *Cryptocurrency Mining Profitability Calculator*. Accessed: 2017. [Online]. Available: <http://www.coinwarz.com/calculators/bitcoin-mining-calculator>
- [27] D. M. Kreps, "Game theory and economic modeling," *J. Public Finance Public Choice*, vol. 9, no. 3, pp. 217–219, Jan. 1991.
- [28] B. Barak, R. Canetti, Y. Lindell, R. Pass, and T. Rabin, "Secure computation without authentication," *J. Cryptol.*, vol. 24, no. 4, pp. 720–760, Oct. 2011.
- [29] P. Gazi, A. Kiayias, and D. Zindros, "Proof-of-stake sidechains," in *Proc. IEEE Symp. Secur. Privacy (SP)*, San Francisco, CA, USA, May 2019, doi: [10.1109/SP.2019.00040](https://doi.org/10.1109/SP.2019.00040).
- [30] *Tendermint*. [Online]. Available: <https://tendermint.readthedocs.io/en/master/>
- [31] *Delegated Proof of Stake*. [Online]. Available: <https://bitshares.org/technology/delegated-proof-of-stake-consensus/>
- [32] D. Stone, "Delayed blockchain protocols," 2018, *arXiv:1804.06836*. [Online]. Available: <http://arxiv.org/abs/1804.06836>
- [33] C. Decker and R. Wattenhofer, "Information propagation in the bitcoin network," in *Proc. IEEE P2P*, Trento, Italy, Sep. 2013, pp. 1–10.
- [34] R. Pass, L. Seeman, and A. Shelat, "Analysis of the blockchain protocol in asynchronous networks," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn. (EUROCRYPT)*, Paris, France, 2017, pp. 643–673.



KAIYU WANG received the M.S. degree in computer science and technology from the Harbin Institute of Technology, where he is currently pursuing the Ph.D. degree. His research interests include blockchain consensus protocol, peer to peer internet security, and trusted computing.



YAN WANG (Senior Member, IEEE) received the D.Eng. degree from the Harbin Institute of Technology. He is currently a Professor with the Department of Computing, Macquarie University, Australia. He has published a number of research papers in international conferences and journals. His research interests include trust management, social computing, service computing, recommender systems, and cyber security.



ZHENZHOU JI received the Ph.D. degree in computer science from the Harbin Institute of Technology, in 2000. He is currently a Professor with the Harbin Institute of Technology. His research interests include computer architecture, parallel processing computer, and computer network security. He was the Vice-Chairman of the Computer Architecture of China Computer Federation (CCF).

...