PoW Security-Latency and Transaction Rate

Mustafa Doger Sennur Ulukus
Department of Electrical and Computer Engineering
University of Maryland, College Park, MD 20742
doger@umd.edu *ulukus@umd.edu*

***Abstract*—We analyze how secure a block is after the block becomes k-deep, i.e., security-latency, for Nakamoto consensus under an exponential network delay model. We give parameter regimes for which transactions are safe when sufficiently deep in the chain. Next, modeling the blockchain system as a batch service queue with exponential network delay, we connect the security-latency analysis to sustainable transaction rate of the queue system. We modify the selfish-mining attack to hamper the service process and consider its effect on the sustainable transaction rate of the queue.**

I. Introduction

Blockchains allow users to maintain a distributed ledger of accounts. Nakamoto's white paper [1] described a mining procedure of nonces (proof of work, PoW) and a longest chain protocol that enables the users to converge to a single version of the ledger. The protocol requires the users to mine a new block at the tip of their longest chain of blocks which contains a nonce proving the validity of the block.

Some blocks are mined on the same height in the chain due to network delays and create forks that divide the network into two different versions of the ledger. Adversarial activities can create deliberate forks as well that undermine the security of the chain and result in larger latency until a block can be confirmed together with its transactions. The more one waits to confirm a block, the safer the content of the block will be. This is referred to as the *security-latency problem*. The security-latency problem has been extensively analyzed by various different models and protocols [2]–[10]. Recently, [9] suggested new bounds on the security question both in terms of time units and number of blocks mined. Further, [11] showed that PoW protocols are safe under random network delays using rudimentary mining models.

As transactions arrive to the blockchain system, they are included in the next blocks that are to be mined based on the fee they offer to its miner. In this sense, the blockchain systems can be modeled as a priority-queue where transactions enter a queue and compete to be included in the next block. Different batch service queues were modeled for blockchain analysis [12]–[15]. Since each mined block experiences a network delay until the other miners can observe it, which decreases the rate of service of transactions, [16] models a batch service queue system that accounts for the network delay and gives the stability condition for the queue system as well as a numerical method to find the steady state of the system.

In this paper, we first give bounds on the security-latency question, i.e., how secure a block is after a certain number of blocks are published, where we re-model the delay with exponential distribution and consider a rigged model first introduced in [6]. Then, we show that the security level can be parameterized in terms of the proportion of mining rate to the network delay which we call κ. Next, the security latency analysis is connected to the queue model via the stability condition of the batch service queue system which enables us to reinterpret the number of transactions a blockchain system can sustain as a function of on the block size, network conditions and κ. We also consider several queue service attacks and their effect on sustainable transaction rate. Although we consider a first-come first-served queue model, this model could be interpreted as a queue system, where we only consider transactions in the service that have a high priority, i.e., it is a first-come first-served queue model for transactions that pay a sufficient fee. Otherwise, it does not make sense to consider the stability condition as the low-fee transactions are too many to have a stable system. This priority model is inspired by the two-class queue model of [12].

Our findings imply that for given network conditions, i.e., a block size b, honest proportion α, security violation probability threshold $\overline{p}$, increasing k-block confirmation rule allows the mining rate to increase which in turn increases the sustainable transaction rate. This adjustment of confirmation rule and mining rate first drops the expected transaction confirmation time until a point where a minimum is achieved and further increasing k-block confirmation rule starts to increase the expected transaction confirmation time, i.e., a trade-off emerges.

II. System Model

We assume the reader has familiarity with blockchain protocols, specifically those that employ PoW and longest chain rules. We denote our system model as $\mathbb{B}(\alpha, b, \mu_1, \mu_2, \lambda, k)$. A block in this model is considered to be valid if it contains a valid nonce and at most b transactions all of which have to be semantically correct (we assume they are). The transactions arrive with an exponential rate of λ to the mempools of the nodes to be included in the next blocks and are served based on first-come first-served principle. We assume a transaction arrives to all mempools at the same time (each honest node has its own mempool based on the longest chain they observed). We abstract out all semantic details of validity of transactions and block contents together with a valid nonce as a Poisson process with mining (arrival) rate of μ_2. Each block at height h that is mined by an honest node is subject to a network delay of $t_{hi} \in [0, \Delta_h]$ before it becomes available to any other

honest node i, where $\Delta_h \sim Exp(\mu_1)$ where Δ_h are i.i.d. In this model, a single adversary is assumed to be controlling all $\beta = 1 - \alpha$ mining power (fully-coordinating adversaries) and the network delay as long as $t_{hi} \in [0, \Delta_h]$, and ties are broken in adversary's favor. A transaction in this model is said to be confirmed according to a k-block confirmation rule if it is part of the longest chain of an honest view and k-deep, i.e., there are at least $k-1$ blocks mined on top of it.

Models with constant maximum delays were proven to be secure against adversarial behaviour with $\beta < \frac{1-\beta}{1+(1-\beta)\lambda\Delta}$ in the long-run [4]. Even though we do not find an ultimate fault tolerance result for exponential delay, we give loose conditions under which the model is secure and rigorously calculate an upper bound for safety violation probability.

Further, we note that a slightly different version of this system model without any adversarial behavior and maximum exponential delay, i.e., $t_{hi} = \Delta_h$ for all i was studied as a two-step batch service queue in [16]. Table I shows some frequently used variables throughout this paper to ease the navigation of the paper for the reader.

Parameters	Definitions
σ	$\frac{\mu_1}{\mu_1+\mu_2}$
ρ	$1-\sigma$
σ'	$\frac{\mu_1}{\mu_1+\mu_2\beta}$
ρ'	$1-\sigma'$
$\bar{a}_i$	$\alpha\rho^i + \beta \cdot \mathbb{1}_{i\leq 2}$
$\bar{b}_i$	$\alpha\rho^i + \beta \cdot \mathbb{1}_{i\leq 1}$
κ	$\frac{\rho}{\sigma} = \frac{\mu_2}{\mu_1}$

TABLE I: Frequently used notations.

III. Rigged Private Attack

We consider a rigged jumper model and private attack for adversarial strategy, to be defined next and denoted as $\overline{\mathbb{B}}(\alpha, b, \mu_1, \mu_2, \lambda, k)$. Consider the genesis block, which is at height $h = 0$ and available to all nodes at time $t = 0$ and called zeroth jumper block. At time $t > 0$, the honest node i is trying to mine a new block on top of the genesis block that contains $\min(b, |Q_i(t)|)$ transactions, where $Q_i(t)$ is the mempool at time t observed by honest node i. After the first honest block on $h = 1$ is mined, say at time t_1, all other honest blocks mined by other honest nodes on the same height $h = 1$ between t_1 and $t_1 + \Delta_1$ are converted to an adversarial block, i.e., they are rigged. As a result, b_1, which we call first jumper, that was mined first on $h = 1$, will enter the view of all other honest nodes at time $t_1 + \Delta_1$ and will be the only block at this height for all of them since they will have to mine on $h = 2$ after $t_1 + \Delta_1$. We assume the same conversion of honest blocks for $h = 2, \ldots$ which we call rigged jumper model which has a single honest block on each height. It is proven in [6] that *private attack* is the best attack to cancel a transaction after it is confirmed under the condition that all honest blocks are on different heights (we denote this condition in short as AHBODH) for bounded Δ-delay model. Next, we restate this theorem proven in [6] for exponential network delay.

Theorem 1 (Guo-Ren [6]) *Under AHBODH, if any attack succeeds in violating a transaction's safety then the private mining attack also succeeds in violating that transaction's safety.*

The safety violation probability $\overline{p}(\alpha, \mu_1, \mu_2, k)$ of $\overline{\mathbb{B}}(\alpha, b, \mu_1, \mu_2, \lambda, k)$ is an upper bound on the safety violation probability $p(\alpha, \mu_1, \mu_2, k)$ of $\mathbb{B}(\alpha, b, \mu_1, \mu_2, \lambda, k)$ which follows from Theorem 1 and the fact that adversary is given the advantage of rigged blocks. Note that, in this model, all mempools during $[t_h + \Delta_h, t_{h+1}]$ are the same. Further, as any honest block mined during $[t_h, t_h + \Delta_h]$ is rigged, the mempools observed by honest nodes are effectively the same all the time, which is denoted as $Q(t)$. We call the mining process of building blocks with a valid nonce, controlled by rate μ_2, as the block-generation process. The network delay process after a block is mined, which is controlled by rate μ_1, is called blockchain-building process.

A. Security-Latency Analysis

We are interested in security-latency analysis of a certain high priority transaction tx which arrives at the mempool at time τ, i.e., we would like to find the safety violation probability which is the probability that tx is discarded after it is confirmed under k-block confirmation rule. Here, for simplicity, we assume at the time of arrival of tx, $|Q(\tau)| < b$. This assumption can be interpreted as a transaction that has very high priority and will enter the next block. The rigorous analysis done in this section can be extended for the case where $|Q(\tau)| \geq b$ [17].

B. Pre-Mining Gain

We start by analyzing the lead of the adversary at time τ, which is the difference between the longest chain and the longest honest chain just before the transaction arrives at the system. As it was done in [5], [6], [8], we assume τ is large, thus we model the lead as the stationary distribution of an extended birth-death process of the rigged model. The steady state distribution of the following extended birth-death Markov chain transition matrix will be equal to adversaries' lead [17]:

$$P = \begin{bmatrix} \alpha\sigma & \alpha\sigma\rho + \beta & \alpha\sigma\rho^2 & \alpha\sigma\rho^3 & \ldots \\ \alpha\sigma & \alpha\sigma\rho & \alpha\sigma\rho^2 + \beta & \alpha\sigma\rho^3 & \ldots \\ 0 & \alpha\sigma & \alpha\sigma\rho & \alpha\sigma\rho^2 + \beta & \ldots \\ 0 & 0 & \alpha\sigma & \alpha\sigma\rho & \ldots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}. \quad (1)$$

Lemma 1 (Ramaswami [18]) *Steady state of P can be found recursively using*

$$\pi_i = \frac{\pi_0 \bar{b}_i + \sum_{j=1}^{i-1} \pi_j \bar{a}_{i+1-j}}{1 - \bar{a}_1}, \quad i \geq 1, \quad (2)$$

where $\pi_0 = 3 - \sigma^{-1} - \alpha^{-1}$, and the definitions of $\bar{a}_i$ and $\bar{b}_i$ are provided in Table I.

The lead distribution is denoted as Π with $P(\Pi = i) = \pi_i$.

C. Confirmation Interval

At time τ, which is assumed to be large, target tx enters the system and will be included in the next honest block by assumption (e.g., high priority). Since τ is large enough and the lead is calculated as a steady state scenario, i.e., as an initial condition, we assume that the process described from $t = 0$ until τ is independent from what follows next. Starting from τ until the time that tx is confirmed by the honest miners, i.e., the block containing tx becomes k-deep in all honest views, is called confirmation interval. Denoting the mining time of the kth jumper block after τ as τ_k and exponential delay of the kth jumper block as Δ_k, confirmation interval spans $[\tau, \tau_k + \Delta_k]$.

Lemma 2 *The number of adversarial (including rigged) arrivals during the confirmation interval, denoted as S_k, has the following distribution,*

$$P_{S_k}(s) = \alpha^k \sigma^k \beta^s \sum_{n=0}^{s} \binom{k-1+n}{n} \binom{k-1+s-n}{s-n} \left(\frac{\rho}{\beta}\right)^n \quad (3)$$

D. Post-Confirmation Race

The block containing target tx will be k-deep at $\tau_k+\Delta_k$ and be confirmed. At the same time, a conflicting block with tx' at the adversarial chain will be $(\Pi + S_k)$-deep. If $\Pi + S_k \geq k$, then the adversary can publish its private chain and cancel tx. Else, the race enters the post-confirmation phase, where adversarial deficit is $D = k - \Pi - S_k$. In this part of the race, adversary has to make up for its deficit D in order to discard the confirmed transaction tx. If at any time after $\tau_k + \Delta_k$, the adversary's private chain is able to catch the longest honest chain containing tx, it can publish its chain to cancel tx. We give a method in [17] that expresses $\overline{p}(\alpha, \mu_1, \mu_2, k)$ explicitly in terms of the distributions of Lemmas 1 and 2.

Lemma 3 $\lim_{k\to\infty} \overline{p}(\alpha, \mu_1, \mu_2, k) \to 0$ *iff* $\alpha > 1/(2-\kappa)$.

Proposition 1 $\overline{p}(\alpha, \mu_1, \mu_2, k) = \overline{p}(\alpha, 1, \kappa, k)$.

Proposition 1 essentially means that the relationship between μ_1 and μ_2 has to be kept constant to keep $\overline{p}(\alpha, b, \mu_1, \mu_2, k)$ at a certain level, hence κ is a safety parameter. In other words, given α, k and a security violation threshold $\overline{p}$, the maximum κ possible, denoted as $\overline{\kappa}$, can be found from the analysis provided in [17], i.e., $\overline{\kappa} = f(\alpha, k, \overline{p})$ for some function f. We do not need to find f explicitly, since our main goal is to establish a relationship between μ_1, μ_2 and security, which is done in Proposition 1. To find an explicit upper bound, we resort to a more simple rigged model originally proposed by [6] where adversary is even more (strictly) powerful than that modeled in rigged jumper model of [8] during the post-confirmation race. At each mining event after $\tau_k + \Delta_k$, we throw a coin: with probability α the block is honest and is converted to an adversarial one if any other block is mined before the block can be published, which happens with probability ρ. Hence, a block becomes and remains honest with probability $\alpha\sigma$. The race between the adversarial and the honest chains at this time can be represented with this coin toss model similar to [6] denoted as $\tilde{M}$, which is distributed geometrically [19, Eqn. (7.3.5)],

$$P(\tilde{M} = i) = \left(1 - \frac{1-\alpha\sigma}{\alpha\sigma}\right)\left(\frac{1-\alpha\sigma}{\alpha\sigma}\right)^i. \quad (4)$$

Lemma 4 *Given mining rate μ_2, honest fraction α, delay rate μ_1 and confirmation depth k, a confirmed transaction cannot be discarded with probability greater than,*

$$p(\alpha, \mu_1, \mu_2, k) \leq \tilde{p}(\alpha, \mu_1, \mu_2, k) = P(\Pi + S_k + \tilde{M} \geq k) \quad (5)$$

Note that using the simple rigged model where a block remains honest with probability $\alpha\sigma$, $\lim_{k\to\infty} \overline{p}(\alpha, \mu_1, \mu_2, k) \to 0$ is trivially satisfied if $\alpha\sigma > \frac{1}{2}$, which can be restated as $\alpha > (1+\kappa)/2$. Further, $\tilde{p}(\alpha, \mu_1, \mu_2, k) = \tilde{p}(\alpha, 1, \kappa, k)$ can be proved using the same ideas as Proposition 1. Our security-latency analysis in this section implies that one can consider μ_2 and μ_1 as dependent parameters when designing a blockchain with a certain security level. Moreover, network delay μ_1 is necessarily a function of the block size b, i.e., $\mu_1 = g(b)$ for some function g, which depends on the network [20]. Thus, given b, α, k and a security violation threshold $\overline{p}$, one can choose the maximum mining rate as,

$$\overline{\mu}_2 = \overline{\kappa}\mu_1. \quad (6)$$

Note that the mining rate should decrease with the increasing block size b to keep the security of the transactions in a block at a certain level since a valid nonce protects the whole block together with all its transactions. The proportional relationship of $g(b)$ and μ_2 shown in (6) is a side effect of the network delay that depends on the block size b, however, it does not consider the security of all b transactions in a single block since $\overline{p}$ analyzed in this section considers the probability that a transaction is discarded. However, adversary can potentially discard more than one transaction (up to b transactions if they happen to be in the same block) with the same attack which is successful with probability less than $\overline{p}$. In this sense, a designer should consider $\overline{p}$ as a security threshold for b transactions while determining μ_2. Finding f rigorously is a complex task, however, one can use bisection method on (5) to find the maximum possible $\tilde{\kappa}$ with $\tilde{p}(\alpha, 1, \tilde{\kappa}, k) < \overline{p}$ which necessarily satisfies $\overline{p}(\alpha, 1, \tilde{\kappa}, k) < \overline{p}$. Since $\tilde{\kappa} \leq \overline{\kappa}$ for a given $\overline{p}$, picking $\tilde{\mu}_2 = \tilde{\kappa} g(b)$ is an approximate solution to the maximum mining rate $\overline{\mu}_2$ where $\tilde{\mu}_2 < \overline{\mu}_2$.

IV. Queue Analysis

To consider the effect of rigged jumper model, private attack on the maximum rate λ the blockchain system can sustain, we resort to the queue analysis in [16]. Fig. 1 describes the Markovian queue process of [16]. Transactions arrive at the mempool-queue, $Q(t)$, with a rate of λ, waiting to be included in the blockchain in a first-come first-served basis. All honest nodes observe the same mempool-queue at any time t, i.e., $Q(t)$. During block-generation process, the nodes try to mine a block with a valid nonce. When a valid nonce is found

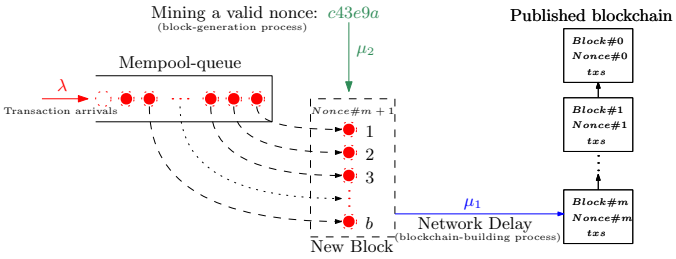


Fig. 1: Queue model.

at any time t, this block will contain the first $\min(b, |Q(t)|)$ transactions from $Q(t)$. After the block is formed with a valid nonce, this block will be added to blockchain according to the blockchain-building process and no other new block is created until the newly mined block is shared with other miners. The steady state behaviour of this Markovian queue process was given in an open form with an iterative formula. Here we only restate main stability conditions and refer the interested reader to [16], [21]. Fig. 2 describes the state transition relation of this system which has the $GI/M/1$ form. Necessary and sufficient condition for the positive recurrence is provided next.

Theorem 2 (Li-Ma-Chang [16]) *The Markov process Q of $GI/M/1$ type is positive recurrent iff*

$$\lambda < \frac{b\mu_1\mu_2}{\mu_1+\mu_2}. \tag{7}$$

Let us denote the maximum λ sustainable for $\mathbb{B}(\alpha, b, \mu_1, \mu_2, \lambda, k')$ and for $\overline{\mathbb{B}}(\alpha, b, \mu_1, \mu_2, \lambda, k)$ as $\lambda(\alpha, b, \mu_1, \mu_2)$ and $\overline{\lambda}(\alpha, b, \mu_1, \mu_2)$, respectively.

Proposition 2 $\lambda(1, b, \mu_1, \mu_2) = \frac{b\mu_1\mu_2}{\mu_1+\mu_2} - \epsilon$.

A. Blockchain Design

In Proposition 1 we established that the relationship between μ_1 and μ_2 has to be kept constant to keep $\overline{p}(\alpha, \mu_1, \mu_2, k)$ at a certain level and κ is a safety parameter. Hence, from now on, we assume μ_2 was chosen as $\overline{\kappa}\mu_1$ at the design stage to keep the security at a certain level. As a result, the stability condition in Theorem 2 and Proposition 2, i.e., under no adversarial attack on queue process, can be re-expressed as

$$\lambda < bg(b)\frac{f(\alpha, k, \overline{p})}{(1+f(\alpha, k, \overline{p}))}. \tag{8}$$

The relation in (8) gives a blockchain designer a means to analyze critical blockchain parameters such as κ, α, k, b and their implications on sustainability and security. In other words, once a desired security level is determined and network conditions are known, the maximum sustainable transaction arrival rate for a blockchain can be obtained using (8).

Note that the way we derive this equation is by assuming that μ_2 was chosen as $\overline{\kappa}\mu_1$ at the design stage to keep security at a certain level. However, this condition does not take into account any adversarial behavior on the queue process that might try to hamper the progress of the blockchain system. Next, we briefly discuss two different private attacks and their effects on sustainability.

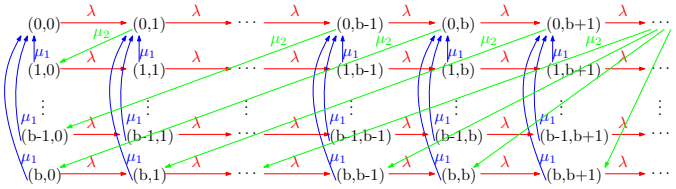


Fig. 2: State transition relation.

B. Queue-Service Attacks

A simple attack is the case where adversary deviates from the protocol and avoids publishing anything at all. A double-spending attack with adversarial private chain is an example for this type of attack assuming the adversary stops the private attack after a successful double spending.

Proposition 3 $\overline{\lambda}(1, b, \mu_1, \mu_2) = \lambda(1, b, \mu_1, \mu_2)$.

Proposition 4 $\overline{\lambda}(\alpha, b, \mu_1, \mu_2) = \overline{\lambda}(1, b, \mu_1, \mu_2\alpha)$.

As a result of Proposition 4 we obtain,

$$\overline{\lambda}(\alpha, b, \mu_1, \mu_2) = bg(b)\frac{\alpha\overline{\kappa}}{1+\alpha\overline{\kappa}}. \tag{9}$$

Notice that in private attack, the goal is to double spend and the model assumes a private chain that is not published until the attack is successful. As a result, private attack only partially slows down the queue process as adversary is essentially not publishing anything. In fact, by slightly changing the private attack strategy and publishing the private chain with empty blocks whenever the lead satisfies some condition, adversary can significantly reduce the service rate of the queue, which we describe next.

Let h_u denote the length of the longest public chain and h_v denote the length of the longest private chain where $h_v \geq h_u$. Let us call "honest mining time" as the time an honest block is mined and "honest publishing time" as the time an honest block is shared with all other honest nodes where the adversary delays all honest block publications with maximum allowed delay. The blocks mined during $[t_h, t_h + \Delta_h]$ are ignored as they all are on the same height h. Consider a strategy, which we call *queue-service attack*, where the adversary mines a private chain with empty blocks as follows: 1) After an honest mining time, t_{h_u}, if there is no private chain, adversary tries to mine on the same height as this lastly mined honest block. If a valid nonce is found at time $t \geq t_{h_u}$ by the adversary, the adversary publishes its block at $\max(t, t_{h_u} + \Delta_{h_u})$ in order to nullify that honest block and starts a private chain on top of its recently published block. 2) At every honest publishing time, $t_{h_u} + \Delta_{h_u}$, if $h_v \geq h_u$, adversary publishes the part of its private chain until (including) the block on height h_u. As the ties are broken in adversary's favor and adversarial blocks do not suffer network delay, the adversarial block will be favored by everyone. Adversary continues mining its private chain.

The algorithm above is a simple variation of the selfish mining attack described in [22], where we slightly change the algorithm since adversary controls network delay and ties are

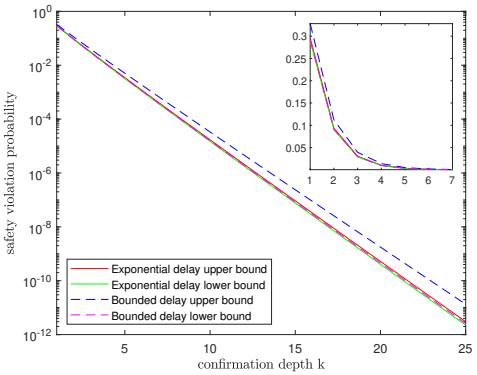


Fig. 3: $\overline{p}$ for exponential and bounded delay, $\alpha = 0.90$.

broken in its favor. Although the original goal of selfish mining attack is to increase the revenues, here we assume that the adversarial blocks are empty in order to hamper the service process of the queue system.

Lemma 5 *Under queue-service attack,* $\alpha' = \frac{\pi'_{-1}}{1-\pi'_{-1}\rho'}$ *proportion of the blocks are honest in the long-run, where* $\pi'_{-1} = 1 - \alpha^{-1} - \sigma'^{-1}$.

We know that the model follows a block generation process with rate $\mu_2\alpha$ and α' proportion of the mined blocks remain on the longest chain and the rest are replaced by empty adversarial blocks. Hence, the maximum λ sustainable under the queue-service attack can be expressed approximately as,

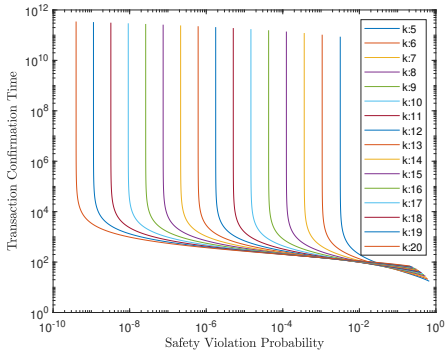
$$\lambda \approx bg(b)\frac{\alpha'\alpha\overline{\kappa}}{(1+\alpha'\alpha\overline{\kappa})}. \tag{10}$$

Notice here that, unlike Proposition 4, the result above is not a rigorous claim since the arrival times of the honest blocks (that are not nullified) in the long-run do not necessarily follow an exponential distribution. As we assume all ties are broken in adversary's favor and adversarial blocks do not suffer any network delay, the results above are extremely pessimistic.

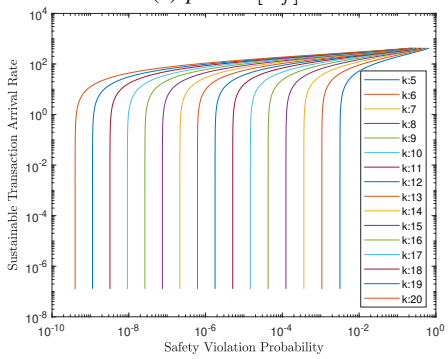
V. Numerical Results

We first compare the security-latency results of our model with that of the existing bounded-delay model in the literature [6], [8]. We present the results for Bitcoin (BTC) parameters, where we choose $\mu_2 = 1/600$, $\Delta = 10$ for bounded delay, and $\mu_1 = \frac{\ln 10}{4}$ as the 90th percentile delay for block propagation is around 4 seconds [23]. Fig. 3 displays the comparison of two models for $\alpha = 0.9$, where it is clear that using an exponential model gives tighter results for upper and lower bounds. The results are not surprising as the bounds in [8] are derived by taking each honest propagation delay as Δ (maximum possible), whereas we take Δ_h (maximum possible). More details of the models compared in the lower and upper bounds and more numerical results are provided in [17].

Next, in Fig. 4, we present the relationships of safety violation threshold $\overline{p}$ with confirmation rule k, security parameter κ, expected transaction confirmation time $E[C_f] = \frac{k}{\mu_2}$ and maximum sustainable transaction arrival rate λ (under no queue attack) for $\alpha = 0.9$ where we choose $b = 4500$



(a) $\overline{p}$ vs $E[C_f]$



(b) $\overline{p}$ vs λ

Fig. 4: $\overline{p}$ effect on (a) $E[C_f]$, (b) λ, for $\alpha = 0.90$.

(corresponds to approximately 1 MB of BTC block [24]) and $\mu_1 = \frac{\ln 10}{4}$. We note that the shape of the curves in Fig. 4 implies that for a given k-block confirmation rule, block size b and network delay rate μ_1, one gains almost nothing in terms of safety violation probability by decreasing safety parameter κ (and hence the mining rate) in the region where graphs are vertical whereas sustainable transaction rate decreases rapidly and the transaction confirmation time increases significantly. Currently, under the 6-block confirmation rule and the current mining rate of BTC, we get a safety violation upper bound of $\overline{p} = 0.00118$ and a maximum sustainable transaction rate of $\lambda < 7.478$. Under no adversarial mining, this rate drops to $\lambda < 6.732$ whereas under the queue-service attack, it further drops down to $\lambda < 5.985$. By simply increasing the safety violation upper bound to 0.002 from 0.00118, BTC safety parameter can be increased to $\kappa = 0.02$, this in turn allows up to ≈ 50(≈ 45 and ≈ 40 under queue attacks) transactions per second and transaction confirmation time of 9 minutes.

The results are even more interesting when one changes 6-block confirmation rule to 10-block confirmation rule, where by choosing $\overline{p} = 0.00118$, we get $\kappa \approx 0.1$, which corresponds to a transaction confirmation time of $\approx$ 3 minutes and up to $\approx$ 235 ($\approx$ 213 and $\approx$ 191 under attacks) sustainable transactions per second. Increasing the k-block confirmation rule for a given safety threshold allows even more gains in terms of the sustainable transactions per second until a point where we have to trade-off this gain with transaction confirmation time. This phenomenon is observable in Fig. 4a where the tail of the graphs cross each other in right bottom.

REFERENCES

[1] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. https://bitcoin.org/bitcoin.pdf, March 2008.

[2] J. Garay, A. Kiayias, and N. Leonardos. The bitcoin backbone protocol: Analysis and applications. In *EUROCRYPT*, April 2015.

[3] R. Pass and E. Shi. Rethinking large-scale consensus. In *IEEE CSF*, August 2017.

[4] A. Dembo, S. Kannan, E. Tas, D. Tse, P. Viswanath, X. Wang, and O. Zeitouni. Everything is a race and Nakamoto always wins. In *ACM CCS*, November 2020.

[5] J. Li, D. Guo, and L. Ren. Close latency-security trade-off for the Nakamoto consensus. In *ACM AFT*, November 2021.

[6] D. Guo and L. Ren. Bitcoin's latency–security analysis made simple. In *ACM AFT*, September 2023.

[7] M. Doger and S. Ulukus. Security bounds for bitcoin under network delay. In *IEEE ISIT*, June 2023.

[8] M. Doger and S. Ulukus. Refined bitcoin security-latency under network delay. Available online at arXiv2212.01372.

[9] S.-J. Cao and D. Guo. Trade-off of security, latency, and throughput of the Nakamoto consensus. Available online at arXiv2312.05506.

[10] P. Gazi, L. Ren, and A. Russell. Practical settlement bounds for proof-of-work blockchains. In *ACM CCS*, November 2022.

[11] S. Sankagiri, S. Gandlur, and B. Hajek. The longest-chain protocol under random delays. *Stochastic Systems*, 13(4):457–478, 2023.

[12] S. Kasahara and J. Kawahara. Effect of bitcoin fee on transaction-confirmation process. *Journal of Industrial and Management Optimization*, 15(1):365–386, 2019.

[13] I. Malakhov, A. Marin, and S. Rossi. Analysis of the confirmation time in proof-of-work blockchains. *Future Generation Computer Systems*, 147:275–291, 2023.

[14] Q.-L. Li, Y. Ma, J.-Y. Ma, and Y. X. Chang. Information theory of blockchain systems. In *Combinatorial Optimization and Applications*, pages 443–454. Springer Nature Switzerland, 2024.

[15] S. Ricci, E. Ferreira, D. S. Menasche, A. Ziviani, J. E. Souza, and A. B. Vieira. Learning blockchain delays: A queueing theory approach. *SIGMETRICS Perform. Eval. Rev.*, 46(3):122–125, January 2019.

[16] Q.-L. Li, J.-Y. Ma, and Y.-X. Chang. Blockchain queue theory. In *Computational Data and Social Networks*. Springer International Publishing, 2018.

[17] M. Doger and S. Ulukus. Transaction capacity, security and latency in blockchains. Available online at arXiv2402.10138.

[18] V. Ramaswami. A stable recursion for the steady state vector in markov chains of m/g/1 type. *Communications in Statistics. Stochastic Models*, 4(1):183–188, 1988.

[19] S. M. Ross. *Stochastic processes*. John Wiley & Sons, Inc., second edition, 1996.

[20] Y. Shahsavari, K. Zhang, and C. Talhi. A theoretical model for block propagation analysis in bitcoin network. *IEEE Transactions on Engineering Management*, 69(4):1459–1476, 2022.

[21] Q.-L. Li, J.-Y. Ma, Y.-X. Chang, F.-Q. Ma, and H.-B. Yu. Markov processes in blockchain systems. In *Computational Social Networks*. Springer International Publishing, 2019.

[22] I. Eyal and E. G. Sirer. Majority is not enough: Bitcoin mining is vulnerable. *Communications of the ACM*, 61(7):95–102, July 2018.

[23] DSN Bitcoin Monitoring. https://dsn.tm.kit.edu/bitcoin/. Accessed: 2023-12-30.

[24] Y Charts. ycharts.com/indicators/bitcoin_average_transactions_per_block/. Accessed: 2023-12-30.