Hindawi
Security and Communication Networks
Volume 2021, Article ID 9549602, 18 pages
https://doi.org/10.1155/2021/9549602

Research Article

Understanding Block and Transaction Logs of Permissionless Blockchain Networks

Hwanjo Heo [1,2] **and Seungwon Shin** [1]

[1]*Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Republic of Korea*
[2]*Electronics and Telecommunications Research Institute (ETRI), Daejeon, Republic of Korea*

Correspondence should be addressed to Seungwon Shin; claude@kaist.ac.kr

Received 15 April 2021; Revised 18 June 2021; Accepted 15 July 2021; Published 4 August 2021

Academic Editor: Wenjuan Li

Public blockchain records are widely studied in various aspects such as cryptocurrency abuse, anti-money-laundering, and monetary flow of businesses. However, the final blockchain records, usually available from block explorer services or querying locally stored data of blockchain nodes, do not provide abundant and dynamic event logs that are only visible from a live large-scale measurement. In this paper, we collect the network logs of three popular permissionless blockchains, that is, Bitcoin, Ethereum, and EOS. The discrepancy between observed events and the public block data is studied via a noble analysis model provided with the soundness of measurement. We share our key findings including a false universal assumption of previous mining-related studies and the block/transaction arrival characteristics.

1. Introduction

Since the inception of Bitcoin, the first peer-to-peer distributed ledger system invented by Nakamoto [1] in 2009, many blockchain systems have undergone development in the public. Ethereum [2] has started its mainnet in 2015 to enhance the vision of blockchain by featuring the idea of smart contracts; it is recognized as the first decentralized computing platform for decentralized applications (dApps). Public blockchains have evolved afterwards to provide better privacy [3], scalability [4, 5], and financial service specialization [6] and even for a particular application environment such as Internet of Things [7]. Among others, EOS [8] has been recognized for its technical endeavor of pursuing a scalable dApp platform with a governance model via Delegate Proof-of-Stake (DPoS) consensus; it is, however, often denounced for not being truly decentralized [9].

Information recorded in permissionless (A blockchain system is permissionless if no permission is required to access or participate in the network.) blockchains, beyond their original utility of being transaction ledgers, is publicly available and commonly studied in the literature. For example, block explorer services [5, 10–16] have emerged to provide transaction information stored in blockchains with additional analytics and user-friendly interfaces. Similarly, historical block and transaction data have become a popular subject not only for economic aspects of cryptocurrencies such as cryptocurrency abuse [17], anti-money-laundering [18], and monetary flow of businesses [19] but also to evaluate the security of underlying mechanisms of blockchains [20–22].

However, historical blockchain data is limited to provide block and transaction records in *historical view*, that is, the eventual ordered set of transaction records, which are only the final product of the consensus mechanism among distributed blockchain network nodes. Live block and transaction records exchanged by the network nodes are more abundant and dynamic so that observing them in *measurement view* plays an important role in understanding the systematic aspects of permissionless blockchain systems.

Figure 1 describes an illustrative time series of transaction and block arrival events in two different view models. Historical blockchain data only provides a *historical view* in

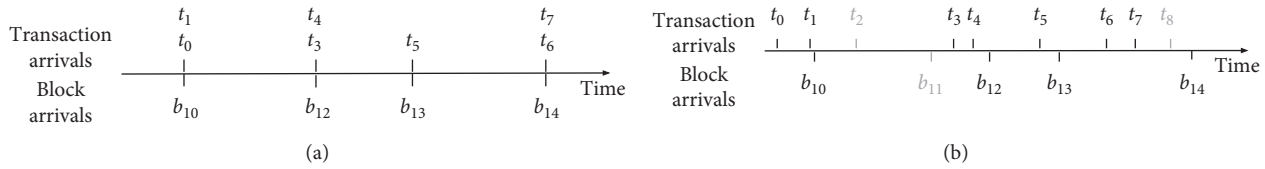

Figure 1: Two view models of blockchain events; t_i are transaction arrival events and b_i are block arrival events; grey-colored events are only observable in the measurement view. (a) Historical view. (b) Measurement view.

which transactions are flattened, that is, appear as if simultaneously arriving at the time of block arrival, and only those included in the main chain blocks are visible. For example, the arrival times of t_0 and t_1 can be only bound to the timestamp of b_{10} in the historical view as there are no separate timestamps for transaction records. Transactions and blocks can also be stale, that is, not included in blocks or not part of the main chain, respectively, or not even explicitly advertised if observed under the hood. For example, the stable block b_{11} which is not the part of the main chain cannot be found on the historical record; the stale transaction t_2 that fails to be included in a block will not be recorded, either. The *measurement view* can provide a rich and dynamic transaction and block arrival event logs, on the other hand.

In this paper, we conduct an in-depth analysis of large-scale block and transaction arrival event measurement of the three popular blockchains: Bitcoin, Ethereum, and EOS. The block and transaction logs, which are not visible from the main chain block data, are investigated to reveal our findings that shed light on the understanding of how permissionless blockchain systems work. Specifically, we make the following major contributions:

(i) Transaction and block arrival events of the three popular permissionless blockchains are measured in eight geographically dispersed locations. Our measurement is underpinned by a noble analysis model and supporting results of measurement soundness.

(ii) We perform a quantitative analysis on the discrepancy between observed events and the public block data. We further study the block discrepancy in terms of blockchain forks to show that the assumption universally adopted by mining-related research does not hold for Ethereum.

(iii) We analyze transaction arrival processes to find that they indeed exhibit LRD (Long-Range Dependence). We provide analytical distributions that fit best the permissionless blockchains we study.

2. Background

2.1. Blockchain. The blockchain stores transactions in the unit of blocks. Transactions are defined differently per blockchain system; it commonly represents a single transfer of cryptocurrency values. Ethereum further defines that a transaction is an action initiated by an externally owned account to modify the state stored in the blockchain network [23]. An EOS transaction is the set of one or more such actions [24].

A miner (or a block producer) can add a block to the chain by publishing it over the network hoping that her published block will become the part of the canonical chain (We also call it the main chain in this paper.); upon successful mining (or block production), she gets rewarded in its native cryptocurrency.

We call the blockchain permissionless if anyone can participate in the mining process. Consequently, there can be temporarily more than one candidate block to be included in the main chain and we call such cases blockchain forks. The main chain refers to the longest [1], heaviest-subtree [25], and irreversible [26] chain for Bitcoin, Ethereum, and EOS, respectively.

Bitcoin and Ethereum adopt PoW (Proof-of-Work) consensus mechanism where mining of a block is performed by solving the computationally demanding cryptopuzzles. On the other hand, EOS adopts DPoS (Delegate Proof-of-Stake) consensus mechanism where the set of block producers, who later produce scheduled blocks in turn, is elected by EOS stakeholders. Table 1 summarizes the three blockchains we study in this paper.

2.2. Research Questions. Our study aims to shed light on the understanding of various aspects of blockchains obscured from a historical view. Specifically, we have focused on three research questions:

RQ1: Are network-advertised transaction logs consistent with the transactions included in main chain blocks?

(i) (Preview on our findings) No. Basically, the blockchain transaction arrival processes are more LRD (Long-Range Dependent) than those of the blocks (Section 5.1). There are unadvertised transactions that are directly inserted by miners, implying privatization of transaction fees. Advertised transactions also may end up being stale, that is, not included in blocks, mostly due to conflict transactions which instead get placed in blocks (Section 4.2).

RQ2: Do miners behave rationally (or selfishly) to maximize their profit? (Do miners endorse earliest propagated blocks in the event of blockchain forks?)

(ii) (Preview on our findings) No. Bitcoin fork events now have become too rare (0.04%), implying that

Table 1: Blockchain summary.

Blockchain	Consensus mechanism	Block time	Transactions per second (all-time high)	Balance model	Transaction fee	Peer-to-peer management		
						Discovery	Max # peers	(Outgoing)
Bitcoin	PoW	~10 minutes	7	UTXO	Yes	Random from seeder list	125	(8)
Ethereum	PoW	~10 seconds	15	Account	Yes	Kademlia-like	50	(16)
EOS	DPoS	0.5 seconds	4,000	Account	No (staking)	—	—	

selfish mining is not the preferred strategy for Bitcoin miners. On the other hand, the majority of Ethereum miners add blocks to their own mined blocks rather than the earliest propagated ones that are mined by other pools; this contradicts the unarguably adopted assumption of previous studies (Section 4.1).

RQ3: Do blockchain nodes effectively get away from partitioning attacks by virtue of information redundancy from multiple peers? (Are they still vulnerable to partitioning attacks?)

(iii) (Preview on our findings) Yes. The reference implementations of PoW blockchains have increased the number of (outbound) peer connections and, consequently, the up-to-date nodes benefit from redundant block advertisement from 14 to 46 peers for Bitcoin and Ethereum, respectively (Section 5.2); tampering with a small number of peers will not successfully partition or make a node unsynchronized for minutes [27].

2.3. Analysis Model. Although the three blockchains operate differently (as summarized in Table 1) in many ways, they can be commonly modeled with the following:

(i) Blocks are produced by miners (or block producers) and are broadcast to the network by peer connections. A block b is uniquely identified with its *blockhash* and refers to its parent block b_p (i.e., $\text{prev}(b) = b_p$). The arrival time of b at a measurement point m (i.e., $\text{arrival}(b)^m$) is the arrival time of b from one of m's peers who delivers b for the first time. The block b can be included in the main chain (i.e., B_{main}), or it can be stale (i.e., $b \notin B_{\text{main}}$).

(ii) Transactions are composed and propagated by users through a P2P network. A transaction t is uniquely identified with its *txid*. The arrival time of transaction t at a node m (i.e., $\text{arrival}(t)^m$) is the arrival time of t from one of m's peers who delivers t for the first time. The transaction t can be included in block b (i.e., $t \in \text{transactions}(b)$) or be stale (i.e., $t \notin \text{transactions}(b) \forall b \in B_{\text{main}}$).

The set of notations is summarized in Table 2 and the functions used in the definition are listed in Table 3.

Our model comprises the two arrival events, that is, block and transaction arrival events, where a single block arrival event is mapped to zero or more transaction events (i.e., $t_i \in \text{transactions}(b)$) by the function $\text{transactions}(\cdot)$ (Bitcoin coinbase transactions are excluded in our analysis as they are not advertised to the network).

Figure 2 depicts how we analyze our measurements. In studying the discrepancy in block and transaction events (Section 4), the arrival events near the measurement boundaries significantly affect our quantitative analysis. For example, including or excluding a single Bitcoin block around the measurement boundary can produce more than two thousand stale or unobserved transactions. To this end, we crop the measurement to the *observation period* by carefully choosing two anchor blocks, the preamble block b_m and the concluding block b_n, for each dataset of blockchains, so that our measurement satisfies the two following conditions: (i) There is no loss of measurement data (due to the reasons described in Section 3.1) at any of our eight (Due to unreliable peering services of candidate BPs (block producers) of EOS, we have relaxed the conditions to six for EOS.) locations in the time period of one week before the arrival of b_m and one week after the arrival of b_n. (ii) Both b_m and b_n belong to the main chain.

3. Measurement

3.1. Data Collection. We have modified the reference implementations of three popular pieces of blockchain software, Bitcoin Core [28], Go Ethereum [29], and EOS.IO [8], to facilitate logging of all received transactions and blocks from peers. Our monitoring pieces of software are deployed to eight geographically dispersed regions of the Google cloud platform [30]: Asia-East, Asia-South, Europe-North, Europe-West, North America-Northeast, US-Central, US-East, and US-West. Each VM is configured to have two vCPUs, 18.5 GB of memory, and 800 GB of standard persistent disk. Measurements were performed between July 2019 and December 2019 by having our monitors, which are indeed full nodes on the three blockchain mainnets, connected to the peers. As regards EOS, the full node software, that is, *nodeos*, is neither bundled with default seed peers nor facilitated with a peer discovery mechanism. We have manually collected available peers and fed them. Also, as *nodeos* only accepts transaction and block advertisements in In-Sync mode [24], a recent snapshot [31] is inputted to quickly catch up without throttling CPU and network bandwidth by syncing from scratch, that is, the genesis block. Unfortunately, we experienced several problems that resulted in suspension of our monitoring processes. For

Table 2: Transaction and block set notation of our measurement framework.

Notation	Description	Definition
B_{obs}	The set of blocks observed in the observation period	$\{b_i, b_{i+1}, \ldots, b_{j-2}, b_{j-1}\}$ where there is no b_k satisfying $\text{arrival}(b_m) \le \text{arrival}(b_k) \le \text{arrival}(b_i)$ or $\text{arrival}(b_{j-1}) \le \text{arrival}(b_k) \le \text{arrival}(b_n)$
B_{main}	The set of blocks that are included in the main chain	$\{b_{m+1}, b_{m+2}, \ldots, b_{n-1}, b_n\}$ where $b_i = \text{prev}(b_{i+1})$ for all $m < i < n$ and b_m and b_n are the preamble and concluding blocks, respectively.
B_{stale}	The set of observed blocks that are stale	$B_{\text{obs}} - B_{\text{main}}$
B_{unseen}	The set of blocks that are part of the main chain but not observed	$B_{\text{main}} - B_{\text{obs}}$
T_{obs}	The set of transactions observed in the observation period	$\{t_i, t_{i+1}, \ldots, t_{j-2}, t_{j-1}\}$ where there is no t_k satisfying $\text{arrival}(b_m) \le \text{arrival}(t_k) \le \text{arrival}(t_i)$ or $\text{arrival}(t_{j-1}) \le \text{arrival}(t_k) \le \text{arrival}(b_n)$
T_{pre}	The set of transactions observed before the observation period	$\{t_0, t_1, \ldots, t_{i-2}, t_{i-1}\}$ where there is no t_k satisfying $\text{arrival}(t_{i-1}) \le \text{arrival}(t_k) \le \text{arrival}(b_m)$
T_{post}	The set of transactions observed after the observation period	$\{t_j, t_{j+1}, \ldots\}$ where there is no t_k satisfying $\text{arrival}(b_n) \le \text{arrival}(t_k) \le \text{arrival}(t_j)$
T_{stale}	The set of observed transactions that are stale	$T_{\text{obs}} - \bigcup_{b \in B_{\text{main}}} \text{transactions}(b)$
T_{included}	The set of transactions that are observed and included in the main chain blocks	$T_{\text{obs}} \cap \bigcup_{b \in B_{\text{main}}} \text{transactions}(b)$
T_{unseen}	The set of transactions that are included in the main chain blocks but not observed	$\bigcup_{b \in B_{\text{main}}} \text{transactions}(b) - T_{\text{pre}} \cup T_{\text{obs}} \cup T_{\text{post}}$

Table 3: Block and transaction functions.

Function	Description
$\text{blockhash}(b)$	The hash value of block b
$\text{prevhash}(b)$	The previous block hash value of block b
$\text{prev}(b)$	The previous block b_p of block p, i.e., $\text{blockhash}(b_p) = \text{prevhash}(b)$
$\text{height}(b)$	The block height of block b
$\text{miner}(b)$	The miner of block b
$\text{children}(b)$	The set of child blocks of block b, i.e., $\text{height}(b_c) = \text{height}(b) + 1 \forall b_c \in \text{children}(b)$
$\text{transactions}(b)$	The set of transactions included in block b
$\text{arrival}(e)$	The arrival time of event e where e can be either a block or a transaction
$\text{inputs}(t)$	The set of transaction inputs (i.e., vin) of Bitcoin transaction t
$\text{outputs}(t)$	The set of transaction outputs (i.e., vout) of Bitcoin transaction t
$\text{nonce}(t)$	The nonce value of Ethereum transaction t

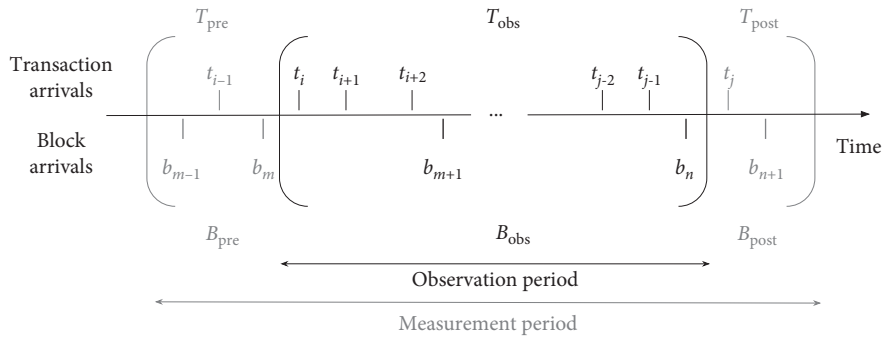

Figure 2: Transaction and block analysis model; grey-colored events are cropped out for measurement point consistency.

example, the Google cloud platform endeavors to detect cryptocurrency-mining-related activities. Once detected, the VM instance immediately gets suspended and is reinstated only if a reasonable explanation is provided.

Furthermore, we have noticed a few software crashes, VM suspensions by other reasons, or temporary lack of disk space causing loss of measurement data. Lastly, a majority of EOS candidate BPs who have provided the peering service have gradually stopped their services and only several are left at the end of our measurement period. To eradicate inaccuracy due to the above problems, our measurement dataset is trimmed as described in Section 2.3. Table 4 summarizes our choice of the two anchor blocks and consequent observation period statistics.

Table 4: Observation period statistics.

Blockchain	Preamble block		Concluding block		Population	
	Block height	Timestamp	Block height	Timestamp	# blocks	# transactions
Bitcoin	587,500	2019-07-28 21:02:28	597,500	2019-10-02 06:22:37	10004	21,454,436
Ethereum	8,290,000	2019-08-05 10:02:10	8,620,000	2019-09-25 19:28:27	353,155	36,665,088
EOS	69,900,000	2019-07-21 21:00:01	75,300,000	2019-08-22 04:08:04	5,402,099	144,198,985

3.2. Measurement Soundness. As public blockchain networks are built on the best-effort serviced overlay network of P2P connections, complete and timed information delivery is not guaranteed. We analyze the completeness of block arrival events, transaction arrival events, and their arrival time accuracy to underpin the soundness of our measurement.

3.2.1. Block Coverage. Blocks can be part of the main chain or they can be stale. We test our observation against the main chain, which is available from a number of block explorer services, to make sure there are no missing blocks.

Table 5 shows the observed number of blocks across our measurement points. The main chain blocks are *all observed*, while stale blocks are partially observed; stale blocks being partially observed is expected as blockchain nodes are not obliged to propagate stale blocks. The stale blocks account for only ~0.04% for Bitcoin and EOS, while ~7.02% of the observed blocks are stale for Ethereum. The stale block probability of Bitcoin has been decreasing [21, 32] and Ethereum's stale block probability is 5 ~ 10% larger than known uncle rates (calculated only with reported uncles) as our measurement captures unreported stale blocks as well [33].

3.2.2. Transaction Coverage. Incomplete transaction measurement causes a subset of transactions which are included in blocks never being observed during measurement. Aggregating transactions observed from different monitors can compensate for this incompleteness. The *unseen* transactions are acquired by subtracting the observed transactions from the transactions included in the main chain blocks.

Figure 3 shows the probability of unseen transactions (i.e., $n(T^{M}_{\text{unseen}})/n(T_{obs})$) as a function of the number of combined measurement points. As combining all measurement point transaction arrivals results in a converged frequency of ~0.05% for Bitcoin and EOS and ~0.5% for Ethereum, it implies that unseen transactions will be still observed even if more monitors are deployed to the network. We further study the unseen transactions in Section 4.2.

3.2.3. Timing Accuracy. Consider an ordered arrival timestamp of block b observed at all measurement points M: $\{ts_i | 0 \le i < n(M)\} = \{\text{arrival}(b)^m | \forall m \in M\}$, where $ts_i < ts_j$ for all $i < j < n(M)$. We evaluate the tightness of our measured timing information with $\Delta(ts_0, ts_1)$ and how far the arrival times can be dispersed among the measurement points with $\Delta(ts_0, ts_7)$, where Δ computes the timewise difference. Figure 4 depicts the distribution of the two

Table 5: Block coverage.

Region	BTC		ETH		EOS	
	Longest	Stale	Longest	Stale	Irreversible	Stale
AS-East	10,000	1	330,000	23,160	—	—
AS-South	10,000	3	330,000	23,160	5,400,000	2,087
EU-North	10,000	4	330,000	23,159	5,400,000	2,094
EU-West	10,000	2	330,000	23,157	5,400,000	2,087
NA-Northeast	10,000	3	330,000	23,160	5,400,000	2,151
US-Central	10,000	1	330,000	23,159	5,400,000	2,085
US-East	10,000	3	330,000	23,160	—	—
US-West	10,000	4	330,000	23,155	5,400,000	2,099

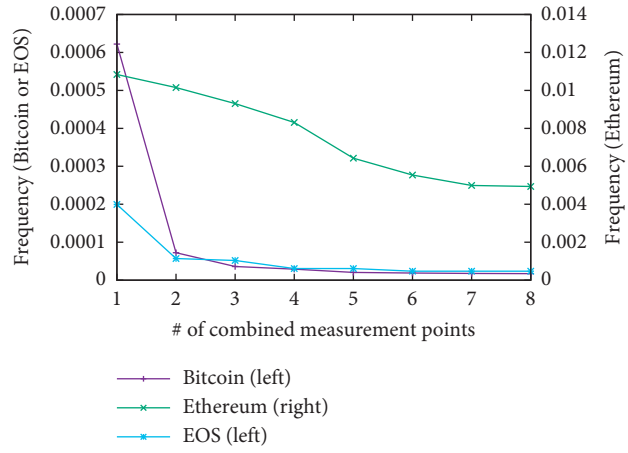

Figure 3: Unseen transaction probability distribution.

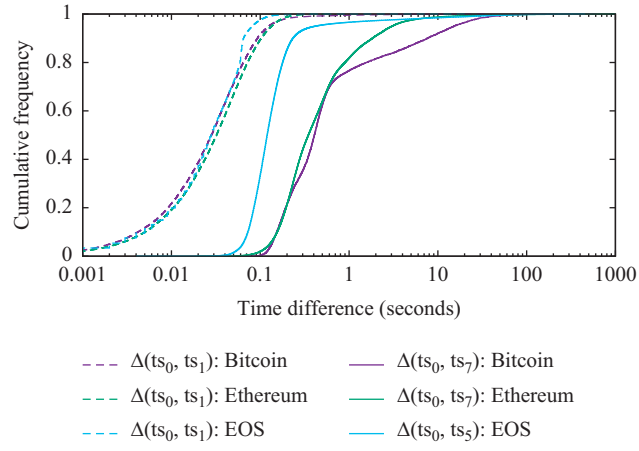

Figure 4: Block arrival time accuracy.

variables for all block arrival events. It shows that the timestamps are tight, as more than 90% of the blocks arrived with a time difference less than 100 ms. Although Bitcoin block arrivals can be delayed by another block time interval (~ 10 minutes), ~ 80% of the blocks are not dispersed by more than one second. EOS blocks show an order-of-magnitude-smaller dispersion in the worst case. The long tails, implying that block arrival times can be very prolonged in a small number of cases, are commonly observed for Bitcoin in the literature [32, 34].

Consider an ordered arrival timestamp of transaction t observed at all measurement points M: $\{ts_i | 0 \le i < n(M)\} = \{\text{arrival}(t)^m | \forall m \in M\}$, where $ts_i < ts_j$ for all $i < j < n(M)$. Figure 5 shows the distribution of $\Delta(ts_0, ts_1)$, $\Delta(ts_0, ts_3)$, and $\Delta(ts_0, ts_7)$ for Bitcoin transactions. Similar to Bitcoin blocks, the long tail shows that the transaction arrival times can be dispersed more than 5 seconds for 5% of the transactions. In the meantime, $\Delta(ts_0, ts_1)$ is tight as more than 99% of the two earliest arrivals are not dispersed more than 1 second. Ethereum and EOS transactions exhibit much tighter $\Delta(ts_0, ts_1)$, showing that more than 98% of the transaction arrivals are less than 100 milliseconds apart.

4. Block and Transaction Discrepancies

In this section, we study the discrepancies in blocks and transactions between the main chain and our observed events.

4.1. Blockchain Fork. As all main chain blocks are observed (i.e., $B_{\text{unseen}} = \varnothing$) through our measurement, the discrepancy in blocks only appears as stale blocks (i.e., B_{stale}). An incident of having stale blocks is also called a *blockchain fork*, where one or more blocks are appended to a different branch of the main chain.

4.1.1. Fork Characteristics and Examples. A nonoverlapping blockchain fork episode e at block height $[m, n)$ is defined by identifying two blocks:

(i) Fork initiating block $b_f \in B_{\text{main}}$ where $\text{height}(b_f) = m$ and $n(\text{children}(b_f)) > 1$

(ii) Fork resolving block $b_r \in B_{\text{main}}$ where $\text{height}(b_r) = n$ and $\text{children}(b_i) = \varnothing$ for all $b_i \in S_n$

$S_n = \{b_i | \text{height}(b_i) = n \text{ and } b_i \notin B_{\text{main}}\}$. Figure 6(a) illustrates our fork episode model.

We define three characteristic variables; (i) its run length $\text{run} - \text{length}(e) = n - m$; (ii) the weight (i.e., the number of stale blocks in the episode), $\text{weight}(e) = \sum_{h=m}^{n} (n(S_h))$; and (iii) the fork degree (i.e., the maximum number of fork branches) $\text{degree}(e) = \max_{m \le h \le n} (n(S_h) + 1)$. Figure 7 shows how characteristic variables are distributed and our findings are summarized as follows:

(i) Due to its long block time (i.e., ~10 minutes) and low stale block probability (i.e., < 0.05%), statistical characterization of Bitcoin fork episodes suffers from a lack of sample events. Even three years of live measurement yield less than a hundred fork episodes as we only expect to observe less than 30 fork events per year.

(ii) Ethereum fork episodes mostly run one block height with a single additional branch. The fork degree (i.e., maximum # of branches) is as high as three for ~ 5% of the episodes and the forks are not immediately resolved, that is, having two or more run lengths, in about 3% of the cases.

EOS fork episodes run as long as 12 blocks, that is, the entire block schedule of a single BP (block producer) [26], as BPs can override the entire block-producing schedule of the preceding BP.

Putting Bitcoin which produces few fork episodes aside, Ethereum and EOS exhibit very different fork episode patterns. Figure 8 shows two example fork episodes of Ethereum and EOS. We have used the commit graph drawing tool [35] to visualize the fork episodes. Figure 8(a) depicts four fork episodes, that is, [#174, #176), [#177, #178), [#183, #184), and [#185, #186), of Ethereum and Figure 8(b) shows two EOS fork episodes, that is, [#290, #292) and [#294, #302). The first fork episode [#174, #176) of Ethereum exhibits the run length of 3, the weight of 4, and the degree of 4 with three nonreported (i.e., thus available only from our measurement) stale blocks. The second fork episode [#294, #302) of EOS exhibits the run length of 9, the weight of 9, and the degree of 2.

4.1.2. Nonearliest Advertised Block Endorsement of Ethereum Miners. Our most interesting finding is that Ethereum miners often do not add blocks to the earliest advertised block during fork episodes. Consider an illustrative fork episode depicted in Figure 6(b). The episode begins as the three blocks (i.e., b_1, b_2, and b_3) are added to the fork initiating block b_0. The immediately following blocks (i.e., b_i and b_j) will have their previous blockhash identifying one of the three previous blocks of the miner's choice. The block b_i, mined by some miner, may be added to the earliest block (i.e., b_1); the other block b_j is mined by a different miner who chooses a different block (i.e., b_2) to add to. To evaluate miners' preference of blocks to which succeeding blocks are added in the event of blockchain forks, we define the conditional probability of miner m_a adding a block to a previous block mined by m_b (by not adding to any of a different miner's block) in fork episodes for all m_a and m_b:

$$\begin{aligned} &\Pr\left(\text{miner}(b_i) = m_b | \text{miner}\left(b_j\right)\right. \\ &\qquad = m_a, \\ &\qquad \text{prev}\left(b_j\right) = b_i, \\ &\exists b_k \colon \text{height}(b_k) = \text{height}(b_i)). \end{aligned} \tag{1}$$

Figure 9(a) shows a histogram of the above probability; m_a (x-axis) adds a block to m_b's block (label) over others in fork episodes. It is apparent that the top ranked miners significantly prefer adding blocks to their own mined blocks over other miners' blocks. Figure 9(b) depicts the same

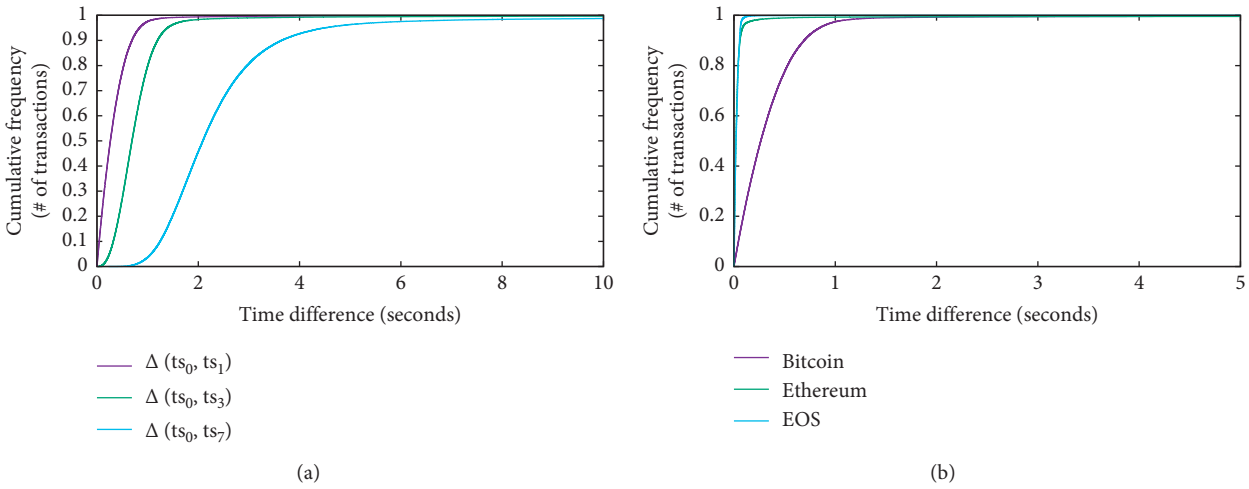

Figure 5: Transaction arrival time accuracy. (a) Transaction arrival time differences of Bitcoin. (b) $\Delta(ts_0, ts_1)$ distribution.

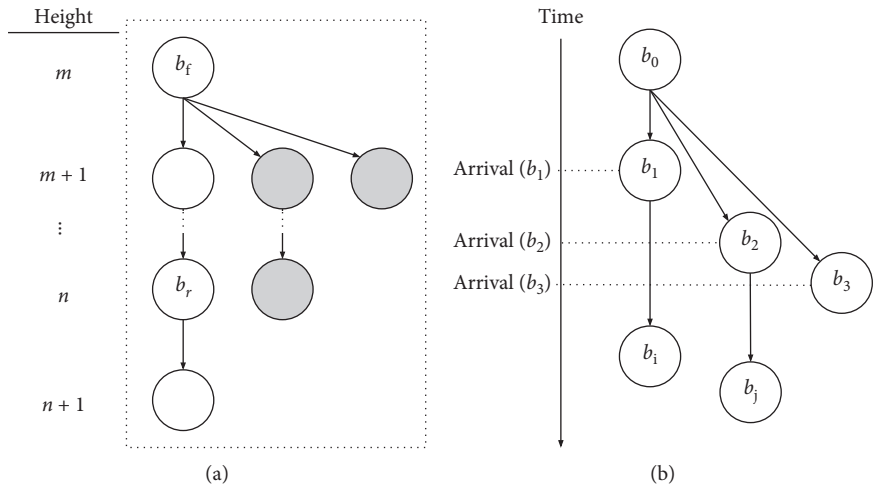

Figure 6: Fork analysis model. (a) Fork episode. (b) Fork resolve example.

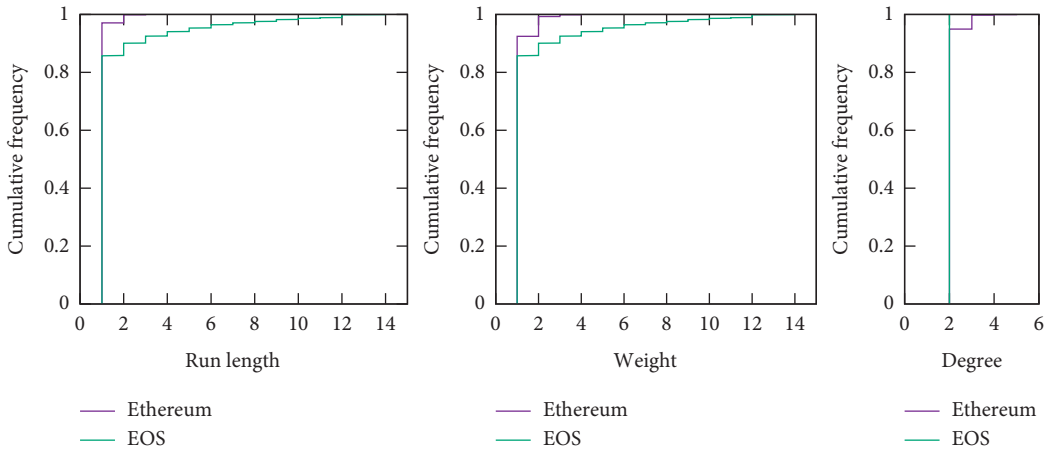

Figure 7: Fork episode characteristics.

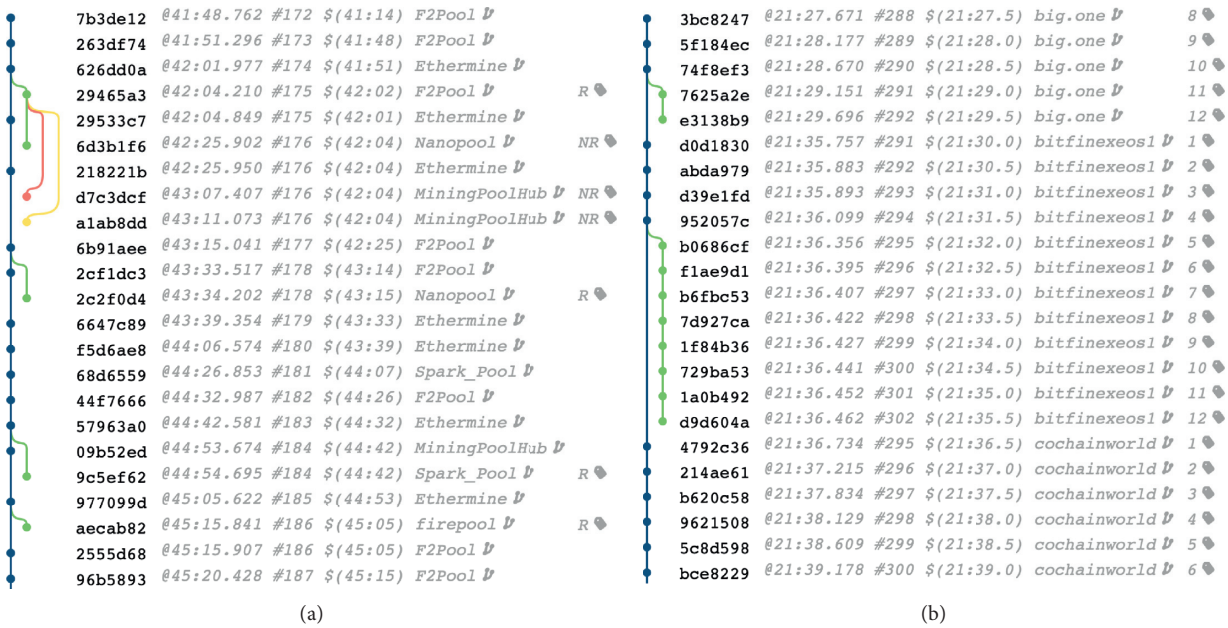

Figure 8: Example fork episodes: each dot represents a block followed by block hash (last six hexadecimals), arrival time, block height (last three digits), block timestamp, and its miner. Each Ethereum stale block is tagged either R (reported uncle) or NR (not reported). Each EOS block is tagged with the block sequence (out of 12) of the corresponding BP's production schedule. (a) Ethereum. (b) EOS.

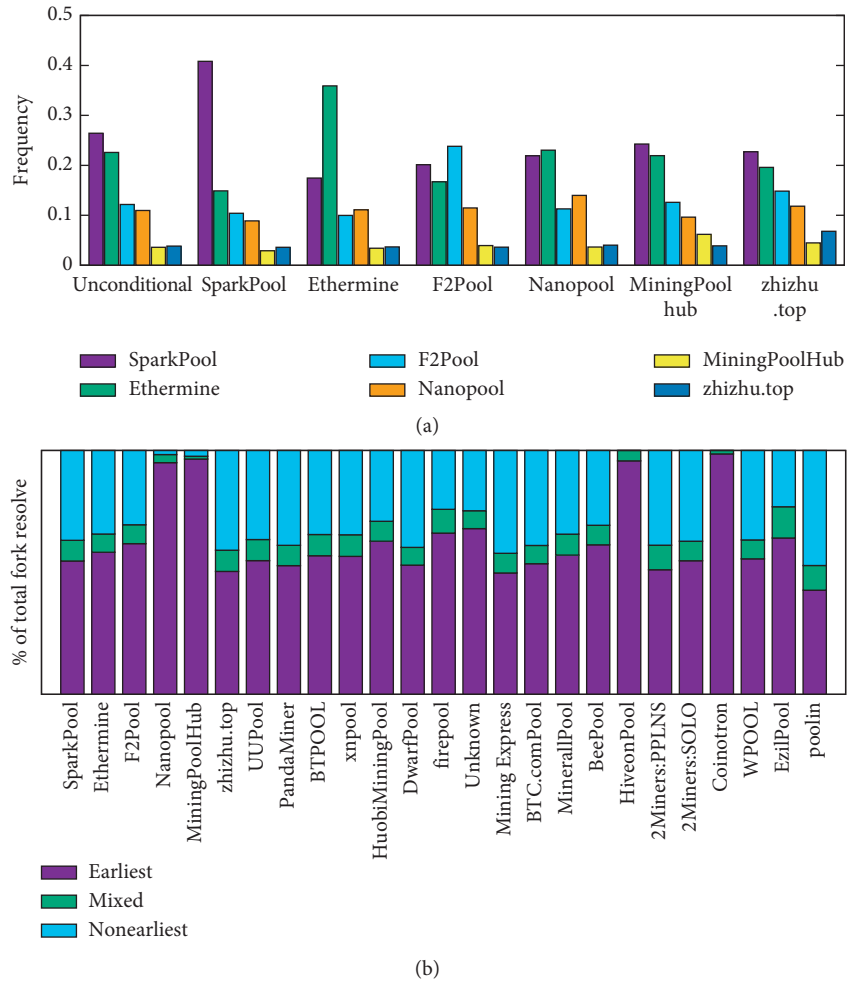

Figure 9: Ethereum fork resolve statistics. (a) Probability of m_a (x-axis) adding a block to m_b's block. (b) Probability of adding to the earliest propagated block.

probability by labeling whether or not the chosen block (by m_a) is the earliest propagated one. The "earliest" label indicates that the block, to which the newly mined block (by m_a) is added, has been observed as the earliest one to arrive congruently among all eight measurement locations (As block arrival times are measured in eight different locations, the arrival order can be conflicting. The "mixed" label represents the conflicting cases.). For example, the top ranked pool *SparkPool* chose the earliest arriving block ~55% of the time, 3409 out of 6234 blocks; among the other 2825 blocks that were added to nonearliest previous blocks, *SparkPool*'s own blocks were chosen over others 1211 times, while other pools' blocks were preferred to *SparkPool*'s only 24 times. In the meantime, *Nanopool* chose the earliest one a majority of the time. *For the top 25 ranked mining pools, only four of them added blocks to the earliest arriving blocks, more than 90% of the time.* This contradicts the earliest propagated block endorsement assumption that is broadly adopted in mining-related research [36–38].

One possible explanation of the phenomenon is that the mining pools adopt their own block preferring strategy as a simple precautionary defense against yet unknown selfish mining pools; the profitability of selfish mining largely depends on the probability of honest pools adding to the attacking pool's branch.

4.1.3. Cut-Tail Fork Pattern of EOS BPs. We find that the EOS fork episodes occur in a way that *BPs are cutting the tail of the previous BP's produced blocks.* EOS adopts the DPoS consensus mechanism [26]; 21 block producers (BPs) are elected by voting and scheduled to produce 12 consecutive blocks in 6 seconds (i.e., 500 ms for each block) in the alphabetical order of their names. Figure 10 shows the numbers of switching (by schedule) events and cut-tail fork events of all immediate BP pairs. We observe a high frequency of such events when *bitfinexeos1* is producing blocks by appending to *big. one*'s blocks; while there are accusations of BP's dishonest behaviors, we believe that the reported *bitfinex*'s performance problem is the compelling reason for the phenomenon [39].

4.2. Unseen and Stale Transactions

4.2.1. Private Transaction Mining. Unseen transactions are characterized by the strong evidence that implies that *the unseen transactions are indeed never advertised to the network*. Table 6 categorizes 95.87% of Ethereum unseen transactions by using the *from* addresses of the transactions (We use prefixes followed by $\star$ in presenting hash values, e.g., address (i.e., pubkey hash) and txid, for brevity. We understand this does not entirely anonymize identity as the readers can verify their results against ours by matching the prefixes.). Public names are retrieved from Etherscan [16]. A miner is specified in the table only if all blocks that include the unseen transactions with the from address are congruently mined by that single miner. It shows that most of the unseen transactions are indeed the block miner's own transactions.

Table 7 shows unseen Bitcoin transactions categorized by the identified reasons. ~ 67% of the unseen transactions are either zero-fee transactions or zero-fee-transaction-dependent transactions. We define a transaction $t \in \text{transactions}(b)$ as a dependent transaction if at least one of input transactions of t is included in the same block b.

Bitcoin Core [28] implementation does not propagate transactions with zero transaction fee. Also, a dependent transaction cannot be mempooled until the transaction to which it is dependent is mempooled; consequently, it cannot be relayed to peers. The Bitcoin Core mempool implementation justifies the inexistence of zero-fee and zero-fee-dependent transactions from our measurement. Existence of a stale transaction which shares the same transaction input with the unseen transaction may imply a possible double-spend attempt, a variant [40] involving private mining of a block with t_i while advertising a conflict transaction t_j such that $\text{inputs}(t_i) \cap \text{inputs}(t_j) \neq \varnothing$; we have observed 27 such unseen transactions without a certainty of attack intentions or if they are not indeed advertised. Lastly, multisig transaction inputs often imply off-chain protocols, where signed transactions are exchanged out-of-band [41] and having OP_RETURN transaction output implies special purpose protocols of Bitcoin, for example, sidechains, where their relationship with miners require further investigation [42]. The presence of unseen transactions, by either private transaction mining or potential out-of-band delivery of transactions, has a subtle security implication. One obvious side effect is privatization of transaction fees; the transaction fees of privately mined ones are exclusively obtainable by the private miner alone, unless the network is fully utilized. If the network is fully utilized, a miner is forced to include her private transactions at the expense of not including other transactions, as the block capacity is limited.

4.2.2. Conflict Stale Transactions. Stale transactions are network-transmitted transactions that are not included in any of the blocks.

Table 8 shows the stale transaction counts observed at each location. The stale transactions account for 0.20%, 3.69%, and 0.47% of Bitcoin, Ethereum, and EOS transactions, respectively; Ethereum transactions had an order of magnitude more chance of being stale than others.

As for public blockchains with transaction fees, that is, Bitcoin and Ethereum, stale transactions are mostly attributable to *conflict*. Table 9 summarizes the identified reasons for transactions being stale. A stale Bitcoin transaction is attributable to *conflict* if there are one or more transactions sharing at least one transaction input (i.e., *vin*).

An Ethereum transaction is attributable to *conflict* if there is another transaction with the same *nonce* from the same account. A widely known reason for two or more transactions being in conflict is transaction replacement such as replace-by-fee [43]. On the other hand, slightly more than 5% of Bitcoin transactions are stale because at least one of their transaction inputs is also stale.

On the other hand, roughly one out of ten stale Ethereum transactions is attributable to improper ordering

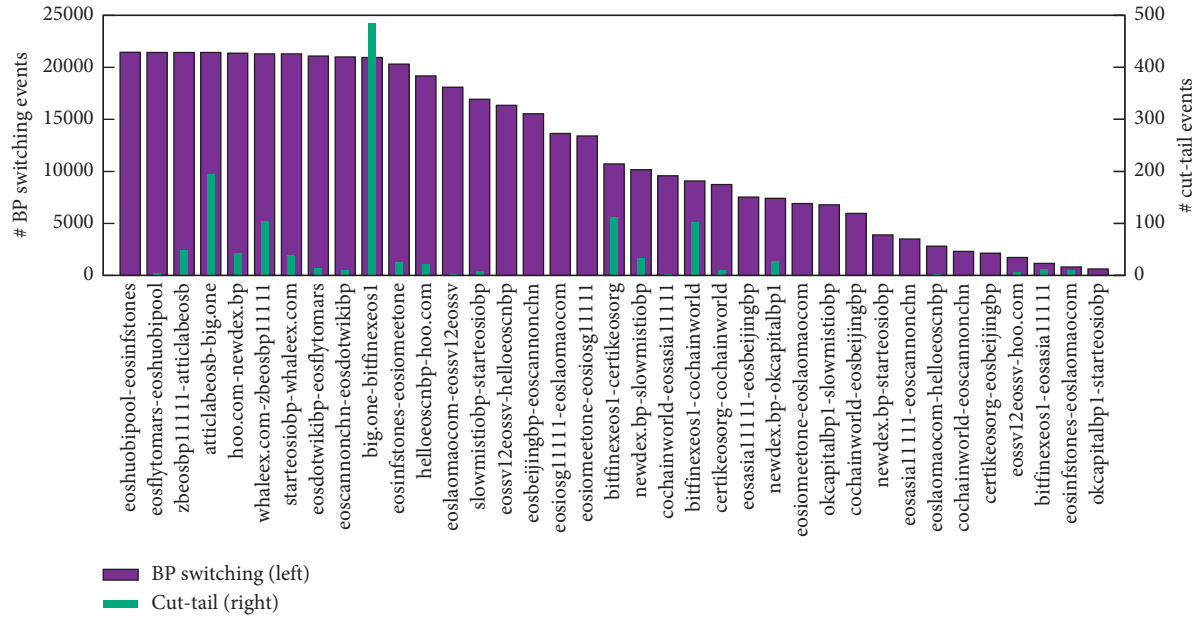

Figure 10: EOS BP switching and cut-tail events.

Table 6: Ethereum unseen transactions.

From address	Public name	Transaction count		Miner
0xea67⋆	Ethermine	128145	(72.42%)	Ethermine
0x2a65⋆	DwarfPool	22955	(12.97%)	DwarfPool
0x8fd0⋆	Unknown	11346	(6.41%)	SparkPool
0x829b⋆	F2Pool	4453	(2.51%)	F2Pool
0x4c54⋆	HiveonPool	1135	(0.64%)	SparkPool
0x99c8⋆	BeePool	468	(0.26%)	BeePool
0xdba7⋆	Unknown	348	(0.20%)	UUPool
0x464b⋆	FKPool	340	(0.19%)	FKPool
0xf687⋆	Unknown	217	(0.12%)	AntPool
0xa801⋆	Unknown	151	(0.09%)	AntPool
0xbe7c⋆	Unknown	126	(0.07%)	AntPool
0xb96f⋆	Unknown	101	(0.06%)	AntPool
Sum		169785	(95.87%)	

Table 7: Bitcoin unseen transactions.

Category	Transaction count	
Zero-fee transaction	148	(50.68%)
Zero-fee-transaction-dependent transaction	48	(16.44%)
Multisig transaction input	43	(14.73%)
Conflict with stale transaction	27	(9.25%)
OP_RETURN transaction output	26	(8.90%)
Sum	292	(100.00%)

of *nonce* (The transaction nonce should be equal to the number of transactions sent from the address.): *nonce* is too high (9.83%) or too low (0.02%). Last but not least, 7.27% of stale transactions are not included in blocks because the account is not sufficiently funded (An Ethereum account should have $balance \geq value + gas \times gas\ price$ to execute the transaction [44].). An Ethereum account can be insufficiently funded when executing a transaction for a number of reasons [45].

4.2.3. EOS Transaction Discrepancy. EOS exhibits unseen and stale transactions far different from those of Bitcoin or Ethereum, that is, the PoW blockchains with transaction fee. EOS software is not bundled with peer discovery; rather the node operator is required to look up for candidate BP's peering services; timely delivery of blockchain data and reliability of the node is dependent on the peering nodes of choice (Indeed, we have experienced service disruption by disconnecting from peers for unknown reasons.). The notion of being conflict in two or more transactions, that is, the most frequent reason for transactions being stale for Bitcoin and Ethereum, is not applicable to EOS (EOS adopts the account model without transaction ordering per account; thus, there should be no conflict transaction outputs or the same account nonce.). On the other hand, EOS requires

Table 8: Stale transactions.

Region	Bitcoin	Ethereum	EOS
Asia-East	41,720	1,170,706	—
Asia-South	41,653	1,194,018	504,277
Europe-North	41,693	1,192,435	470,480
Europe-West	41,668	1,123,680	450,140
North America-Northeast	41,703	1,212,210	588,711
US-Central	41,683	1,212,389	505,160
US-East	41,692	1,186,488	—
US-West	41,672	1,167,627	480,207
Combined	42,023	1,351,847	672,999

Table 9: Identified reasons of stale transactions.

Category	Transaction count	
(a) Bitcoin		
Conflict	39,408	(93.78%)
Stale transaction input	2,307	(5.49%)
Conflict and stale transaction input	308	(0.73%)
Sum	42,023	(100.00%)
(b) Ethereum		
Conflict	1,208,588	(89.24%)
Nonce (skipped)	133,161	(9.83%)
Nonce (used)	251	(0.02%)
Insufficient fund	9,847	(7.27%)
Sum	1,351,847	(99.82%)
(c) EOS		
Fork	12,106	(1.80%)
Others	660,893	(98.20%)
Sum	672,999	(100.00%)

contract users to stake tokens for CPU time that can be deducted by transaction executions. A requested transaction can be rejected, thus being stale, due to the lack of CPU time, or other resources similarly. Replaying the entire state changes of EOS to find out stale transactions due to lack of resources is not possible unless all transaction and state logs of the entire BPs are available.

On the other hand, EOS transactions have two additional attributes which are not available in the Bitcoin or Ethereum transaction structures:

(i) Reference block: every EOS transaction refers to a block, by specifying the block number and hash, not only to prevent replay attacks but also to signal the network that the user's stake is on a specific fork

(ii) Expiration time: EOS transactions have explicit expiration times

If a transaction's reference block is not part of the main chain at the time of execution, the transaction cannot be executed (and thus cannot be included in a block). We have identified that 1.8% of stale transactions apply to this case. To further identify unknown reasons, we consider four timed events that are relevant to transaction execution:

(i) Last irreversible block arrival time arrival (b_{lib}): the arrival time of transaction's reference block. EOS transactions, by default, refer to the last irreversible block (LIB) [26]

(ii) Head block arrival time arrival (b_{head}): the scheduled time of the head block when transaction is arrived

(iii) Transaction arrival time arrival (t): the arrival time of transaction t

(iv) Expiration time exp (t): the expiration time of transaction t

The three time differences, that is, Δ_{exp}, Δ_{incl}, and Δ_{trx}, are investigated to find characteristics of unseen (T_{unseen}) and stale (T_{stale}) transactions over the normal transactions (T_{included}). Figure 11 illustrates the relevant events and time differences of them. Δ_{exp} measures the time difference between the two scheduled (i.e., virtually timed) events; Δ_{incl} and Δ_{trx} are defined by the arrival events to our monitor node. Figure 12 shows the distributions of Δ_{exp}, Δ_{incl}, and Δ_{trx}. We find that unseen transactions have prolonged Δ_{incl}, implying that the delay of newly produced blocks can cause newly generated transactions not promptly delivered because the sync manager is in the *Head Catch-Up* mode as transactions are only transferred in *In-Sync* mode by its design [24]. Stale transactions with relatively small Δ_{trx} can be attributable to problematic dApp implementations referring to non-LIB blocks; the chain instability caused by blockchain fork, mostly due to the cut-tail behavior shown in Section 4.1, can cause transactions being rejected for immature block reference.

However, we do not find any reason attributable to the transaction expiration. Although the unseen transactions tend to have small Δ_{exp}, the expiration times are only as small as 30 seconds (i.e., the default expiration time). Particularly, one or more EOS applications set the expiration time to 180 seconds (other than the default 30 seconds) and more than half of the stale transactions are invoked by those applications.

While *transaction* is the smallest unit of execution in Bitcoin and Ethereum, the unit of contract execution in EOS is *action* [24]. A transaction consists of one or more actions. However, the constituent actions in a single transaction cannot be separately advertised or executed; accordingly, we apply the identical transaction model to EOS without loss of generality.

5. Block and Transaction Arrival Characteristics

5.1. Arrival Processes. All three blockchains we study in this paper are designed to publish capacity constrained blocks in targeted regular time intervals. An incarnation of such design is expected to exhibit a Poisson block arrival process. While it is claimed that Bitcoin block arrivals do not entirely exhibit a homogeneous Poisson process due to its difficult adjustment mechanism [46], transaction arrival processes of blockchains are not studied in the literature to the best of our knowledge. A particularly meaningful step toward understanding the arrival process of events generated by a complex system is examining if the arrival events are independent or exhibiting a correlated structure [47].

The variance-time log-log plots [48] of the block and transaction arrival event time series are depicted in

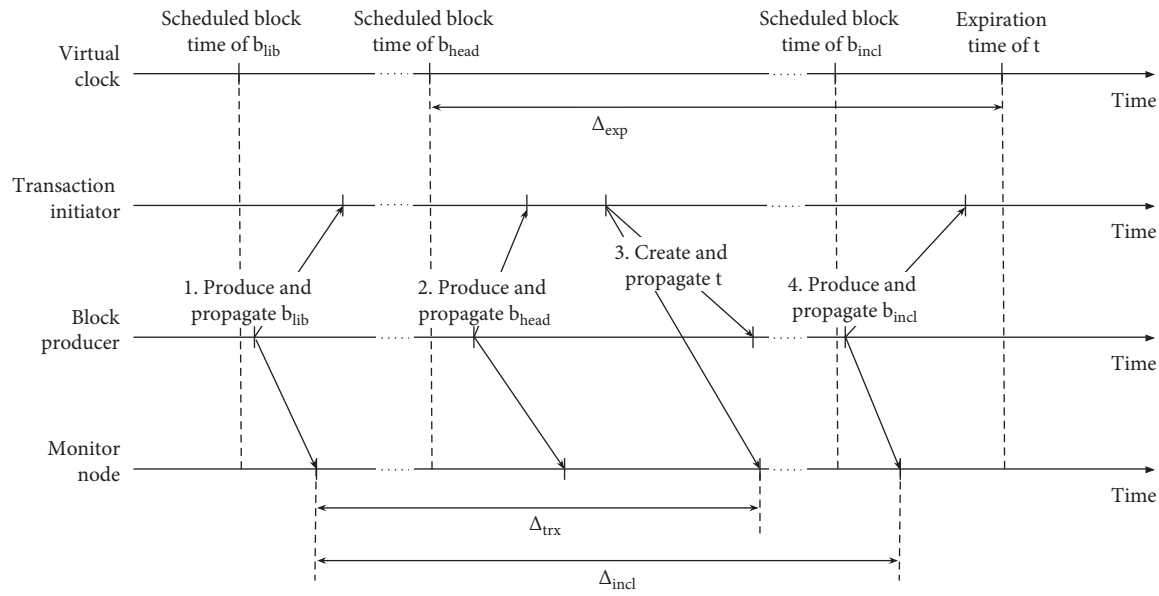

Figure 11: EOS transaction analysis model.

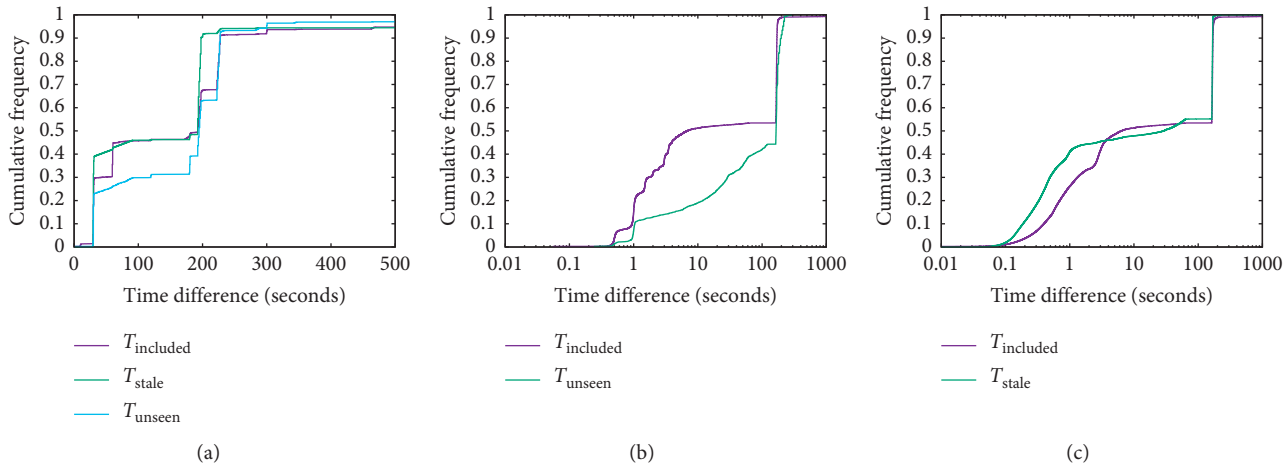

Figure 12: EOS transaction timing analysis. (a) Δ_{exp}. (b) Δ_{incl}. (c) Δ_{trx}.

Figure 13(a). As regards the block arrival processes, Bitcoin block arrivals exhibit a linear slope $\beta = -2/3$. The estimated Hurst parameter $H = 1 + \beta/2 = 4/3$ implies that the process is not a homogeneous Poisson process, but LRD (Long-Range Dependent), contrary to what is expected for a regular interval event generation [46]. Ethereum block arrivals approximate the Bitcoin's except for the high aggregation level due to high stale block rate. EOS block arrivals, in contrast, follow the slope $\beta = -1$ in the low aggregation level as the block generation is indeed regularly scheduled.

Figure 14 shows the variance-time plot of Ethereum and EOS block arrival time series by taking all block arrival events (B_{obs}) and main chain block arrivals only (B_{main}). We confirm that Ethereum block arrivals indeed exhibit a straight line (close to Bitcoin) if excluding stale blocks. However, EOS block arrivals do not exhibit a straight line even if we exclude stale blocks due to delayed burst block arrivals possibly due to cut-tail fork patterns (see Section 4.1).

The transaction arrival processes of all three blockchains appear to exhibit LRD with Hurst parameter $H > 5/6$ as $\beta < -1/3$. To further identify analytical distributions that fit best the empirical ones, the coefficient of variation (σ/μ) and skewness (i.e., the standardized third moment) of transaction interarrival times are computed for every measurement dataset and compared to the relationships expected for the exponential, lognormal, Weibull, and Gamma distributions in Figure 13(b). We find that Bitcoin, Ethereum, and EOS transaction interarrival times are fit best Gamma, Weibull, and lognormal distribution, respectively. Individual datasets are also examined for the fitness of the distribution to find that the empirical distribution fits the analytical distribution

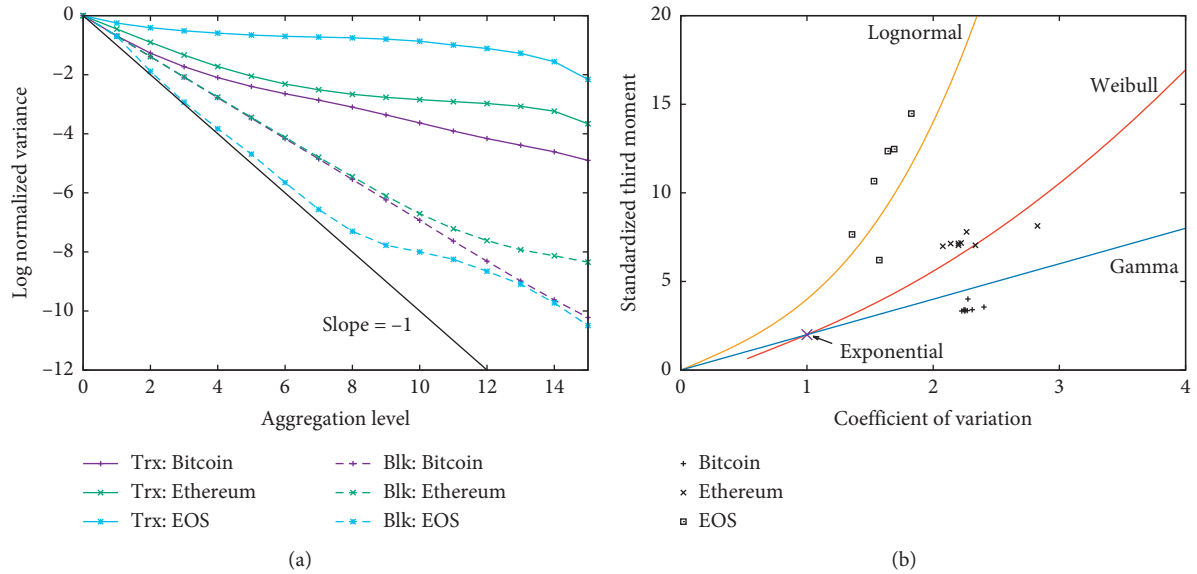

Figure 13: Transaction arrival characteristics. (a) Variance-time plot of block and transaction arrivals. (b) Cov versus skewness of transaction interarrival times.

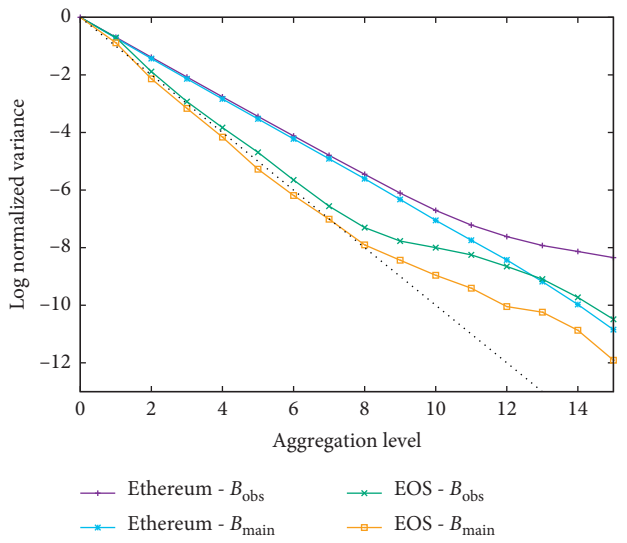

Figure 14: Variance-time plot of block arrivals including stale blocks (B_{obs}) and excluding stale blocks (B_{main}).

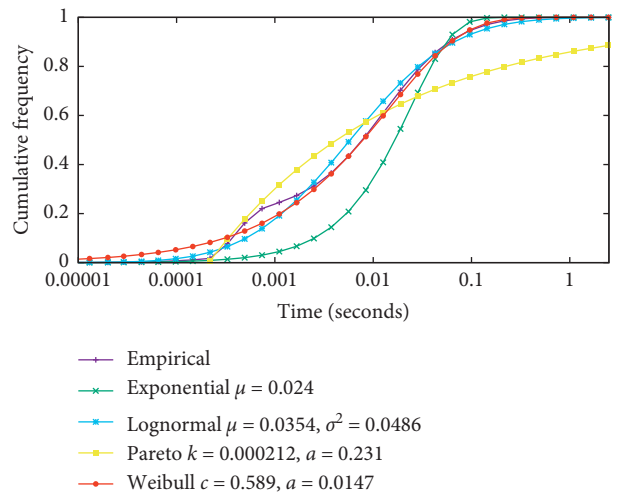

Figure 15: Empirical and fitted analytical distributions of Ethereum transaction interarrival times.

well except for the small interarrival time ranges. For example, Figure 15 depicts the CDF of Ethereum transaction interarrival times observed in US-West. Although the empirical distribution fits Weibull with parameters computed with the maximum likelihood estimator [49], sample points under several milliseconds appear to be unreliable (We are yet unaware of the reason. However, it is known that the time source of commodity operating systems is not accurate and we need an external time source for accurate sub-millisecond timing measurement [50].).

5.2. Advertisement Redundancy. Bitcoin and Ethereum node pieces of software are implemented to maintain a large number of peer connections with multiple random remote nodes. Bitcoin Core can connect up to 125 peers, where 8 of them are outbound connections. Go Ethereum maintains 50 peer connections, where 16 of them are outbound, by default (see Table 1). On the other hand, partitioning attacks are developed to isolate a blockchain node from the others by tampering with all such peer connections. Partitioning attacks usually exploit vulnerable peer management mechanisms of Bitcoin [51] and Ethereum [52]. Such attacks are also examined in the context of ISP's BGP hijacking [53] and a stealthier variant relieving the attacker from the risk of getting exposed [54]. The attacks often rely on implementation vulnerabilities for efficient eclipsing of peers, while known vulnerabilities are getting addressed; for example, the lack of parallel block download sessions causing

temporary denial of blocks (reported in [27] and fixed by recommending three concurrent download sessions in [55]) let attackers successfully partition the node without eclipsing all peer connections [53]. A simple countermeasure of increasing the number of peer connections, on the other hand, is applied to the reference public blockchain implementations [56, 57] to inflate attack complexity.

We investigate the degree of a peer's advertisement redundancy on newly generated blocks and transactions to evaluate the efficacy of partitioning attacks and the countermeasure of increasing the number of peer connections. Figure 16 shows how redundantly newly generated block information is offered by peers. Bitcoin nodes have maintained ~43 peer connections, that is, 8 outbound and additional 35 inbound connections, on average and Ethereum nodes record all 50 peer connections established during the entire observation period.

For each newly generated block, one-third of peers advertised the block header to our Bitcoin node and more than 90% of Ethereum peers signaled availability of the blocks. In consequence, successfully partitioning a node requires compromise of peers proportional to the total number of peer connections. Increasing the number of peer connections will burden the attacker linearly as expected.

Figure 17 shows how redundantly the newly generated transaction information is offered by peers. While one out of three Bitcoin peers also advertises new transactions via INVENTORY message [58], half of the Ethereum peers have sent new transactions to our node in average. Bitcoin and Ethereum blocks are advertised with the block header (rather than the direct delivery of all block contents). Bitcoin transactions are also advertised with INVENTORY message before the whole contents are transferred. However, Ethereum transaction data, that is, the entire contents, are directly sent by peers (Ethereum now has a pooled transaction delivery model in its newer ETH/64 protocol [30].). We also find that there are cases with much less advertised transactions and have identified that they are indeed mostly stale ones; stale transactions are mostly due to conflict (see Section 4.2) and the nodes that already queued the conflict transaction will not relay the stale one.

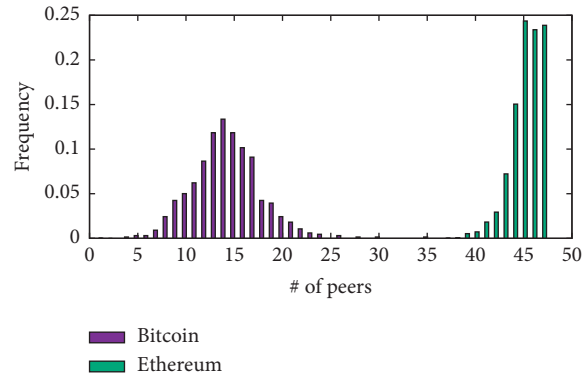

Figure 16: Peer redundancy of block advertisement.

6. Discussion

6.1. Implication on Mining Attacks. Cryptocurrency mining has been a popular subject in assessing the security of Bitcoin; selfish mining [37], block withholding attack [36], and a mixed strategy [38] have been studied. Although it is shown that selfish behavior in mining is always more profitable than being honest, without need for a mutual agreement of not attacking each other [38], we find that selfish mining is not the preferred strategy. All selfish mining strategies will result in blockchain forks, but we observe a very low fork probability (i.e., 0.04%) for Bitcoin. On the other hand, selfish mining on Ethereum has been studied only recently to unanimously claim that selfish mining is also profitable for Ethereum miners; however, it is less profitable than Bitcoin due to its uncle reward system [59]. Our result in Section 4 shows that a majority of Ethereum miners choose to add to their own produced blocks over the earliest propagated blocks. How does this affect the selfish mining strategy?

The selfish mining strategy assumes a parameter γ which defines the fraction of honest miners' mines on the attacker's fork branch (The parameter comes into play in case (f) of [27].). If the attacking pool has a network-wise superior position in block propagation (i.e., $\gamma > 0.5$), the selfish mining strategy is profitable for pools with proportional mining power $\alpha > 25\%$ [37]. Absolute network inferiority (i.e., $\gamma = 0$ due to a defensive mining strategy such as the one presented in Section 4.1) increases the required mining power to 33%; the 33% mining power appears to be significantly harder to achieve in the real world [20].

As for the FAW (Fork-After-Withholding) attack strategy [38], the network capability parameter C defines the probability that an attacker's FPoW (Full Proof-of-Work) block (during infiltration mining) is selected as a main chain block. C affects both the attack reward and whether the miner's dilemma holds or not in the mutual attacking game. As an example case (Section 9 of [38]) of a two-pool game between F2Pool and BitFury, the absolute network inferiority drops C from 0.914 to 0.3 to make it a dilemmatic game again. Unfortunately, miners' own block preferences cannot be directly transferred to a single parameter of γ or C; it calls for a model change from previous mining studies.

6.2. Conflict Transactions and Double-Spending Attacks. Double-spending attacks have been studied in the context of Bitcoin fast payments where zero or only one confirmation is required to complete an offline merchandise transaction [60]; real-world cases where two conflict transactions disparately arrive at different measurement locations may imply such attacks. Table 10 shows the cases where two transactions are in conflict (and thus only one of them is included in a block, i.e., $\text{inputs}(t_i) \cap \text{inputs}(t_s) \neq \varnothing$ and $\exists b \in B_{\text{main}}: t_i \in \text{transactions}(b)$ ($\nexists b \in B_{main}: t_s \in \text{transactions}(b)$ is indisputable given a consistent transaction ledger)) and two different subsets of the monitors have mempooled either one of them. We also have identified 27 unseen transactions implying possible one-confirmation attack variant requiring the private mining (i.e., not advertising the double-spending transaction to the network) in

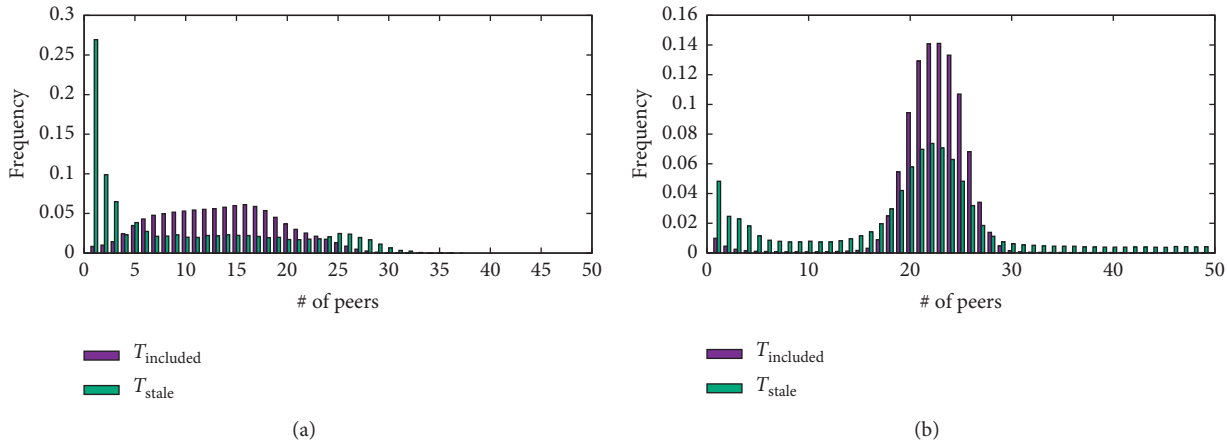

Figure 17: Peer redundancy of transaction advertisement. (a) Bitcoin. (b) Ethereum.

Table 10: Bitcoin disparately mempooled transactions.

# monitors mempooled t_i	# monitors mempooled t_s	Transaction count
7	1	26 (38.24%)
6	2	14 (20.59%)
5	3	10 (14.71%)
4	4	5 (7.35%)
3	5	5 (7.35%)
2	6	3 (4.41%)
1	7	3 (4.41%)
5	1	1 (1.47%)
6	1	1 (1.47%)
Sum		68 (100.00%)

Section 4.2. As for the countermeasure of such attacks by deploying monitors [60], deployment of 8 monitors in the network to detect the attacks would fail at most 27 out of 95 times (~30%), assuming that those 27 unseen transactions we have identified are initially advertised to the network.

6.3. Potential Impact on Blockchain Applications. Ethereum and EOS are the most widely used smart contract platforms fostering various decentralized applications. Bitcoin, which lacks the capability of executing complex smart contract codes, is also recently utilized to execute smart contracts for decentralized applications [50, 61]. Even though our work focuses on the fundamental blockchain mechanisms in terms of transaction/block production and propagation, the impact of our findings on decentralized applications can be foresighted.

According to our blockchain fork results (Section 4.1), the lack of selfish mining in the real world was speculated by a very low stale block rate of Bitcoin network and prevalence of nonearliest block endorsement by Ethereum miners. As profitability of selfish mining encourages miners to behave in a way of producing more blockchain forks, the performance degradation is proportionate to the number of miners who adopt the selfish mining strategy; the key performance metrics affecting the blockchain application performance such as node response time or block delivery time are shown to be noticeably degraded due to the selfish mining [62]. For example, decentralized identities issued or revoked on such blockchains [63, 64] cannot be responsively finalized due to inflated blockchain forks. Our results show that the selfish mining is not prevalent in the real world; thus the application performance impact is overrated, even though it is theoretically shown that miners have every reason to be selfish [38]. As for EOS, we reported cases where a sequence of (up to 12) blocks being stale are often observed in successive mining of two BPs; however, the rate is too low (i.e., 0.04%) to impact the performance of applications (Section 3.2).

Transaction arrival processes being LRD (Section 5.1) pose a technical challenge in provisioning transaction processing scalability especially for Layer-2 blockchain application solutions [65, 66]. For example, Layer-2 rollup solutions require transaction relays, which aggregate individual user transactions of a given blockchain application, deployed in the network often without a priori knowledge of network load. One way of solving the problem is a proper prioritization of transactions based on the application context.

Lastly, blockchain network partitioning can be the biggest concern for blockchain service providers. We observed that the increased numbers of peer connections in blockchain reference implementations effectively provide a higher degree of information redundancy (Section 5.2). However, increased number of peer connections can be exploitable for an attacker who has a denial-of-service attack vector or an invalid transaction initiated by a simple application bug can flood the network [45].

6.4. Limitation. Our measurement study has several limitations. First of all, our results may only capture only a few months of blockchain network operations. In the meantime, the permissionless blockchain pieces of software are in active development; for example, Ethereum now utilizes the pooled transaction delivery [67] instead of unsolicited transaction delivery, which was the only mode at the time of our

measurement. Both cryptocurrency market values and blockchain network utilizations are rapidly increasing and thus miners are getting more motivated to optimize their performance; for example, Bitcoin fork probability has become far smaller than what is reported previously (see Section 4.1).

Second, the blockchain state information is not collected in our measurement. Our analysis is performed based on only available information stored in block and transaction logs. We have utilized an Ethereum archive node [44] to query historical state information (e.g., account balance or account nonce) in Section 4.2. However, such proactive state replication is far more complicated with EOS unless the full node is configured to operate as a producing node and logging the state information; thus, EOS stale transactions could not be fully understood without the missing EOS network state information.

7. Related Work

7.1. Blockchain Data Analysis. Historical transaction data of Bitcoin have often been studied for economic aspects such as cryptocurrency abuse [17], anti-money-laundering [18], and monetary flow of businesses [19]. However, historical blockchain data, usually available from block explorer services, are not sufficient for our research purpose.

7.2. Blockchain Measurement. Researchers have performed blockchain measurement studies to assess the degree of decentralization [20], to investigate demographic information of network nodes [34, 68], to study mining pool behavior [22], and to evaluate security of PoW blockchains [21] against known attacks such as double-spending attack [60] and selfish mining [37]. Our work expands the understanding of previous security evaluations with the key findings acquired from our large-scale transaction and block measurement.

7.3. Blockchain Attacks. Partitioning attacks usually exploit vulnerable peer management mechanisms of Bitcoin [51] and Ethereum [52]. Such attacks are also examined in the context of ISP's BGP hijacking [53] and a stealthier variant relieving the attacker from the risk of getting exposed [54]. Cryptocurrency mining has been a popular subject in assessing the security of Bitcoin; selfish mining [37], block withholding attack [36], and a mixed strategy [38] have been studied. Double-spending attacks have been studied in the context of Bitcoin fast payments [60].

7.4. Block Explorer Services. We have surveyed the popular block explorer services of Bitcoin [5, 10, 12], Ethereum [10, 13, 16], and EOS [11, 14, 15] to see the availability of information, beyond what is recorded in the main chain block data, we have utilized in our study; namely,

(i) Stale blocks/transactions: no Bitcoin and EOS block explorers provide stale block information (A related study [36] has utilized a Bitcoin block explorer service [68] to retrieve stale blocks. However, the service has been terminated in March 2019.). All Ethereum block explorers provide only reported stale blocks (i.e., uncle blocks [23]) (Ethereum provides incentives for uncle stale blocks [44] so that miners include them in their mined blocks for profit.).

No block explorers provide stale transaction information unless one is currently pooled.

(ii) Block/transaction arrival times: all block explorers provide block timestamps that are recorded in the block headers, not the actual times of arrival. Our blockchain fork and gap analysis require measured block arrival times at multiple locations. All block explorers show transaction timestamps identical to the block timestamp to which the transactions belong (Recent Bitcoin transactions have different timestamps than the block timestamps in [4], while the timestamps are only at a granularity of one minute.) [69, 70].

8. Conclusion

We have measured transaction and block arrival events of the three permissionless blockchains. We have devised a noble analysis model and analyzed completeness of our block and transaction logs. Finally, we have studied the discrepancy in measured events and arrival characteristics to share our key findings including a false universal assumption of previous mining-related studies.

Data Availability

The measurement data used to support the findings of this study have not been made available because they contain private information that can be used to identify individual blockchain nodes.

Disclosure

This is an extended and revised version of a conference paper presented in IEEE ICDCS 2021 [41].

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors would like to thank Seungwon Woo, Daegeun Yoon, and Changhyun Lee for helping them with the measurement instrumentation. This work was supported by Electronics and Telecommunications Research Institute (ETRI) grant funded by the Korean Government (21ZR1300, Research on Intelligent Cyber Security and Trust Infra).

References

[1] S. Nakamoto, "Bitcoin: a peer-to-peer electronic cash system," 2019.

[2] V. Buterin, "Ethereum white paper.," 2013, https://ethereum.org/en/whitepaper/.

[3] Monero, "Monero," 2020, https://web.getmonero.org.

[4] bitcoincash.org. https://www.bitcoincash.org.
[5] Blockchain.com. https://www.blockchain.com.
[6] Ripple, "XRP: The best digital asset for global payments," 2020, https://ripple.com/xrp/.
[7] I. Foundation, "Iota," 2021, https://www.iota.org.
[8] "Eosio," 2020, https://eos.io.
[9] S. O'neal, "Eos proves yet again that decentralization is not its priority," 2018, https://cointelegraph.com/news/eos-proves-yet-again-that-decentralization-is-not-its-priority.
[10] Blockchair. https://blockchair.com.
[11] Bloks.io. https://bloks.io.
[12] BTC.com, "Bitcoin block explorer," 2021, https://btc.com.
[13] Ethereum block explorer. https://eth.btc.com.
[14] eosflare.io. Eos block explorer. https://eosflare.io.
[15] EOSPark, "Eos block explorer," 2021, https://eospark.com.
[16] Etherscan. Ethereum explorer. https://etherscan.io.
[17] S. Lee, C. Yoon, and H. Kang, "Cybercriminal minds: An investigative study of cryptocurrency abuses in the dark web," in *Proceedings of the Network and Distributed System Security Symposium*, pp. 1–15, Internet Society, San Diego, CA, USA, February 2019.
[18] M. Möser, R. Böhme, and D. Breuker, "An inquiry into money laundering tools in the bitcoin ecosystem," in *Proceedings of the 2013 APWG eCrime Researchers summit*, pp. 1–14, IEEE, San Francisco, CA, USA, September 2013.
[19] M. Lischke and B. Fabian, "Analyzing the bitcoin network: the first four years," *Future Internet*, vol. 8, no. 1, 2016.
[20] A. E. Gencer, S. Basu, I. Eyal, R. V. Renesse, and E. G. Sirer, "Decentralization in bitcoin and ethereum networks," in *Proceedings of the International Conference On Financial Cryptography and Data Security*, pp. 439–457, Springer, Nieuwport, Curaçao, February 2018.
[21] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the security and performance of proof of work blockchains," in *Proceedings Of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 3–16, Vienna, Austria, October 2016.
[22] C. Wang, X. Chu, and Q. Yang, "Measurement and analysis of the bitcoin networks: a view from mining pools," 2019, https://arxiv.org/abs/1902.07549.
[23] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum Project Yellow Paper*, vol. 151, 2014.
[24] "Transactions protocol eosio developer docs," 2018, https://developers.eos.io/welcome/v2.0/protocol/transactions_protocol.
[25] Y. Sompolinsky and A. Zohar, "Secure high-rate transaction processing in bitcoin," in *Proceedings of the International Conference On Financial Cryptography and Data Security*, pp. 507–527, Springer, San Juan, PR, USA, January 2015.
[26] Block.one, "Consensus protocol," 2018, https://developers.eos.io/welcome/v2.0/protocol/consensus_protocol.
[27] A. Gervais, H. Ritzdorf, G. O. Karame, and S. Capkun, "Tampering with the delivery of blocks and transactions in bitcoin," in *Proceedings Of the 22nd ACM SIGSAC Conference On Computer and Communications Security*, pp. 692–705, Denver, CO, USA, October 2015.
[28] "Github: Bit coin," 2020, https://github.com/bitcoin/bitcoin.
[29] G. Ethereum, "Go ethereum," 2020, https://github.com/ethereum/go-ethereum.
[30] Google, "Google cloud," 2021, https://cloud.google.com.
[31] "How to replay from a snapshot," 2018, https://developers.eos.io/manuals/eos/latest/nodeos/replays/how-to-replay-from-a-snapshot.
[32] C. Decker and R. Wattenhofer, "Information propagation in the bitcoin network," in *Proceedings of the IEEE P2P 2013 Proceedings*, pp. 1–10, IEEE, Trento, Italy, September 2013.
[33] "Ethereum uncle rate," 2020, https://ycharts.com/indicators/ethereum_uncle_rate.
[34] J. A. Donet, C. P. Solà, and J. H. Joancomartí, "The bitcoin p2p network," in *Proceedings of the International Conference On Financial Cryptography and Data Security*, pp. 87–102, Springer, Christ Church, Barbados, March 2014.
[35] G. S. Davies, "Bit-booster - offline commit graph drawing tool," 2016, http://bit-booster.com/graph.html.
[36] N. T. Courtois and L. Bahack, "On subversive miner strategies and block withholding attack in bitcoin digital currency," 2014, https://arxiv.org/abs/1402.1718.
[37] I. Eyal and E. G. Sirer, "Majority is not enough: bitcoin mining is vulnerable," in *Proceedings of the International Conference on Financial Cryptography and Data Security*, pp. 436–454, Springer, Christ Church, Barbados, March 2014.
[38] Y. Kwon, D. Kim, Y. Son, E. Vasserman, and Y. Kim, "Be selfish and avoid dilemmas: fork after withholding (faw) attacks on bitcoin," in *Proceedings Of the 2017 ACM SIGSAC Conference On Computer and Communications Security*, pp. 195–209, Abu Dhabi, UAE, April 2017.
[39] B. Research, "Decentralisation, governance and eos - a lost case?," 2020, https://research.binance.com/en/analysis/eos-governance.
[40] B. Wiki, "Irreversible transactions - finney attack," 2012, https://en.bitcoin.it/wiki/Irreversible_Transactions﹨#Finney_attack.
[41] J. Poon and T. Dryja, "The bitcoin lightning network: scalable off-chain instant payments," 2016.
[42] M. Bartoletti and L. Pompianu, "An analysis of bitcoin op_return metadata," in *Proceedings of the International Conference On Financial Cryptography and Data Security*, pp. 218–230, Springer, Sliema, Malta, April 2017.
[43] "Replace by fee," 2018, https://en.bitcoin.it/wiki/Replace_by_fee.
[44] E. Foundation, "Node and clients," 2020, https://ethereum.org/en/developers/docs/nodes-and-clients/.
[45] H. Heo and S. Shin, "Behind block explorers: public blockchain measurement and security implication," in *Proceedings of the 41th IEEE International Conference On Distributed Computing Systems*, IEEE, Washington DC, USA, In press, Washington DC, USA, July 2021.
[46] R. Bowden, H. P. Keeler, A. E. Krzesinski, and P. G. Taylor, "Block arrivals in the bitcoin blockchain," 2018, https://arxiv.org/abs/1801.07447.
[47] M. E. Crovella and A. Bestavros, "Explaining world wide web traffic self-similarity," Technical Report, Citeseer, Princeton, NJ, USA, 1995.
[48] H. Zhang, Y. Shu, and O. Yang, "Estimation of hurst parameter by variance-time plots," in *Proceedings of the 1997 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, PACRIM. 10 Years Networking the Pacific Rim, 1987-1997*, IEEE, Victoria, BC, Canada, August 1997.
[49] A. Feldmann, "Characteristics of TCP connection arrivals," in *Self-Similar Network Traffic and Performance Evaluation*, K. Park and W. Willinger, Eds., pp. 367–399, Wiley, Hoboken, NJ, USA, 2000.
[50] RSK, "RSK," 2021, https://www.rsk.co.
[51] E. Heilman, A. Kendler, A. Zohar, and S. Goldberg, "Eclipse attacks on bitcoin's peer-to-peer network," in *Proceedings of*

the 24th USENIX Security Symposium, pp. 129–144, Washington, DC, USA, April 2015.

[52] Y. Marcus, E. Heilman, and S. Goldberg, "Low-resource eclipse attacks on ethereum's peer-to-peer network," *IACR Cryptology*, vol. 2018, p. 236, 2018.

[53] M. Apostolaki, A. Zohar, and L. Vanbever, "Hijacking bitcoin: Routing attacks on cryptocurrencies," in *Proceedings of the 2017 IEEE Symposium On Security and Privacy (SP)*, pp. 375–392, IEEE, San Jose, CA, USA, May 2017.

[54] M. Tran, I. Choi, G. J. Moon, A. V. Vu, and M. S. Kang, "A stealthier partitioning attack against bitcoin peer-to-peer network," in *Proceedings of the IEEE Symposium On Security and Privacy (S&P)*, San Francisco, CA, USA, May 2020.

[55] B. I. Proposal, "Bip 0152," 2016, https://en.bitcoin.it/wiki/BIP_0152.

[56] B. Core, "Bitcoin Core Version 0.19.0.1 Release Note," 2016, https://bitcoin.org/en/release/v0.19.0.1.

[57] "Geth v1.9.0," 2019, https://blog.ethereum.org/2019/07/10/geth-v1-9-0/.

[58] "Protocol documentation," 2016, https://en.bitcoin.it/wiki/Protocol\text{_}documentation.

[59] C. Feng and J. Niu, "Selfish mining in ethereum," in *Proceedings of the 2019 IEEE 39th International Conference On Distributed Computing Systems (ICDCS)*, pp. 1306–1316, IEEE, Dallas, TX, USA, July 2019.

[60] G. O. Karame, E. Androulaki, and S. Capkun, "Double-spending fast payments in bitcoin," in *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, pp. 906–917, Raleigh, NC, USA, October 2012.

[61] P. Das, L. Eckey, T. Frassetto et al., "Fastkitten: practical smart contracts on bitcoin," in *Proceedimgs of the 28th {USENIX} Security Symposium ({USENIX} Security 19)*, pp. 801–818, Santa Clara, CA, USA, August 2019.

[62] S. G. Motlagh, J. Misic, and V. B. Misic, "The impact of selfish mining on bitcoin network performance," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 1, pp. 724–735, 2021.

[63] A. Mühle, A. Grüner, T. Gayvoronskaya, and C. Meinel, "A survey on essential components of a self-sovereign identity," *Computer Science Review*, vol. 30, pp. 80–86, 2018.

[64] M. Toorani and C. Gehrmann, "A decentralized dynamic pki based on blockchain," in *Proceedings Of the 36th Annual ACM Symposium On Applied Computing*, pp. 1646–1655, Gwangju, Korea, March 2021.

[65] E. foundation, "Optimistic-rollups," 2021, https://docs.ethhub.io/ethereum-roadmap/layer-2-scaling/optimistic\text{_}rollups.

[66] "Zk-rollups," 2021, https://docs.ethhub.io/ethereum-roadmap/layer-2-scaling/zk-rollups/.

[67] E. foundation, "Ethereum wire protocol," 2019, https://github.com/ethereum/devp2p/blob/master/caps/eth.

[68] S. K. Kim, Z. Ma, S. Murali, J. Mason, A. Miller, and M. Bailey, "Measuring ethereum network peers," in *Proceedings of the Internet Measurement Conference 2018*, pp. 91–104, Boston, MA, USA, October 2018.

[69] Symmetricom, "Delivering sub-microsecond accurate time to linux applications around the world," 2012, http://www.chronos.co.uk/files/pdfs/wps/WP_AccurateTimeForLinuxApplications.pdf.

[70] Ycharts, "Blocktrail blockchain data query service (terminated)," 2019, https://blocktrail.com.