# On the Bandwidth Consumption of Blockchains

Andrei Lebedev[1] and Vincent Gramoli[1,2]

[1]The University of Sydney
[2]Redbelly Network

## Abstract

With the advent of blockchain technology, the number of proposals has boomed. The network traffic imposed by these blockchain proposals increases the cost of hosting nodes. Unfortunately, as of today, we are not aware of any comparative study of the bandwidth consumption of blockchains.

In this paper, we propose the first empirical comparison of blockchain bandwidth consumption. To this end, we measure the network traffic of blockchain network nodes of five blockchain protocols: Algorand, Aptos, Avalanche, Redbelly and Solana. We study the variation over time, differentiate the receiving and sending traffic and analyze how this traffic varies with the number of nodes and validators.

We conclude that the transport protocol is the main factor impacting the network traffic, segregating node roles helps reduce traffic and different blockchains are differently impacted by the network size.

## 1 Introduction

With the advent of blockchain technology, the number of proposals has boomed over the past decade. There exist various layer-1 blockchains offering different guarantees and performing differently at large scale. Recent studies [19, 20] have shown that the performance and robustness of some blockchains is quite far from the promises claimed by their designers. The problem stems mainly from a misunderstanding of these blockchains' underlying networking protocols. The impact of these misunderstandings is so important that Avalanche [31] and Solana [37] were even shown to stop globally when some network messages get delayed [20].

The misunderstanding of these networking protocols also presents economic drawbacks. Modern high-throughput architectures have shifted validator economics from static capital expenditures to variable operational burdens, where egress consumption acts as a primary cost driver. For instance, the prohibitively high egress tariffs of standard cloud providers can render high-performance nodes economically non-viable, potentially incurring monthly costs exceeding US$8,000 for a single validator due to misaligned billing models [21]. This structural inefficiency necessitates a strategic pivot toward bare-metal providers with unmetered allowances to avoid the "hyperscaler egress trap" [36].

As of today, there are no studies comparing the network traffic of blockchains. Even though it is well known that some blockchains (e.g., Solana [37]) favor redundancy (e.g., through erasure coding) to cope with packet losses while others (e.g., Redbelly [12]) adopt different communication patterns between validator and non validator nodes, the bandwidth consumption of these blockchains remain unclear. Understanding the network traffic is, however, crucial to improve performance and robustness of layer-1 blockchains by reducing the network congestion to make them scale or to replicate the data that are key to their robustness.

In this paper, we propose the first empirical comparison of blockchain bandwidth consumption. To this end, we build upon the series of work around the Diablo performance benchmark [19] and the STABL fault tolerance benchmark [20] to measure the network traffic of blockchain network nodes in various situations: (i) while receiving and sending messages; (ii) before, during and after the network receives transactions and (iii) as the network size grows both in terms of nodes but also validator nodes.

Using this black-box approach, we deploy five different blockchain protocols, namely Algorand [18], Aptos [3], Avalanche [31], Redbelly [12], and Solana [37]. We make the following observations:

1. The dominant factors of network traffic are the transport protocol (polling vs WebSockets) and the block propagation strategy (full block download vs. hash comparison) more than the transaction size.

2. Segregating roles between different types of nodes helps reduce the network traffic by reducing network traffic between nodes of different types.

3. Solana network traffic depends on the network size, Algorand and Redbelly network traffic increases with the validator sets and Aptos and Avalanche network traffic increases with both the number of nodes and validators.

The paper is organized as follows. In Section 2, we present the background. In Section 3, we present the experimental settings and our methodology. In Section 4, we present the variety of bandwidth consumption of the five blockchain protocols. In Section 5, we study the distribution of network traffic over different pairs of nodes. In Section 6, we compare the bandwidth consumption before, during and after reception of transactions. In Section 7, we study the impact of the number of nodes and validators on the bandwidth consumption. In Section 8, we study the impact of the sending rate on the bandwidth consumption. We present the related work in Section 9 and we conclude in Section 10.

# 2 Background and Blockchain Networks

In this section, we list the characteristics of each tested blockchain network protocol.

### 2.1 The Algorand network

Algorand [18] is a blockchain protocol that shuffles participants via cryptographic sortition to enhance security. More precisely, it uses Verifiable Random Functions (VRFs) to randomly select participants for specific roles in the consensus execution. In Algorand, nodes communicate with a gossip-based protocol where each node validates each message before relaying it and sends it at most once to each other node [9]. To this end, each node maintains one TCP connection per node in its neighborhood, which offers WebSockets over HTTP.

### 2.2 The Avalanche network

Avalanche, based on the Snowflake consensus protocol [31], is a probabilistic blockchain that requires, by default, a proportion of the nodes that collectively own at least 80% of the total stake to be online. In Avalanche, nodes communicate over TCP and exploit throttling to limit their resource usage. More specifically, messages and connections are rate-limited [4] to cap the amount of CPU, disk, bandwidth, and message handling that other nodes consume. Avalanche uses a dynamic proposer selection algorithm to manage network load and block production. After each parent block, it pseudo-randomly selects an ordered list of potential proposers for the next block height, weighted by stake and using a seed derived from the parent block's height and chain ID. Each proposer is assigned a minimum delay based on their position in the list before they can propose a block. If no proposer acts within the cumulative delay period, any active validator may propose. Blocks within the assigned windows must include valid signatures from the designated proposer.

### 2.3 The Aptos network

Aptos [3] is a leader-based blockchain that uses TCP and builds upon a variant of the *Practical Byzantine Fault Tolerant (PBFT)* consensus protocol [5] featuring a view-change mechanism with a quadratic communication complexity and inherits its cubic communication complexity reached when the leader is faulty or the network is unstable.

It is well-known that leaders act as bottlenecks in leader-based consensus protocols [34], like AptosBFT, and that classic blockchains suffer from redundant dissemination of the same transactions first outside and then within blocks [32, 33]. To cope with these limitations, Aptos features the Quorum Store optimization of Narwhal [13] to decouple metadata ordering from payload dissemination. In addition, Quorum Store is designed for parallel execution by all validators. As outlined in the documentation [7], validators repeat the following steps in parallel: *(1)* Pull transactions from the mempool; *(2)* Arrange transactions into batches based on gas price and select an expiration time for each batch; *(3)* Broadcast batches to all other validators; *(4)* Persist received batches, sign their digests, and send back signatures; and *(5)* Collect signatures from more than $2n/3$ nodes to form a *proof-of-store*. It allows validators to asynchronously broadcast transactions, offloading the leader's network interface during the consensus protocol execution [8].

### 2.4 The Redbelly network

Redbelly Blockchain [12] is a scalable blockchain built on the Democratic Byzantine Fault Tolerant (DBFT) consensus algorithm [11] that is *leaderless* (i.e., non leader-based) and deterministic, and works in a partially synchronous environment. To enhance scalability further, Redbelly uses a collaborative approach, appending a superblock with as many valid proposed blocks as possible. This way the number of transactions per appended block can grow linearly with the number of nodes [12].

Redbelly's nodes communicate using TCP and features a Scalable variant of the EVM, called SEVM. It was shown to perform well under realistic dApps particularly in a large geo-distributed environment when compared to other modern blockchains [33].

### 2.5 The Solana network

Solana [37] is a leader-based blockchain that may fork. In order to determine whether a transaction is committed, Solana requires 30 additional blocks to be appended after the transaction's block. Nodes communicate over the QUIC network protocol [23] to exchange transactions. Nodes split blocks into chunks that they disseminate in a hierarchical structure, called Turbine [1], through UDP.

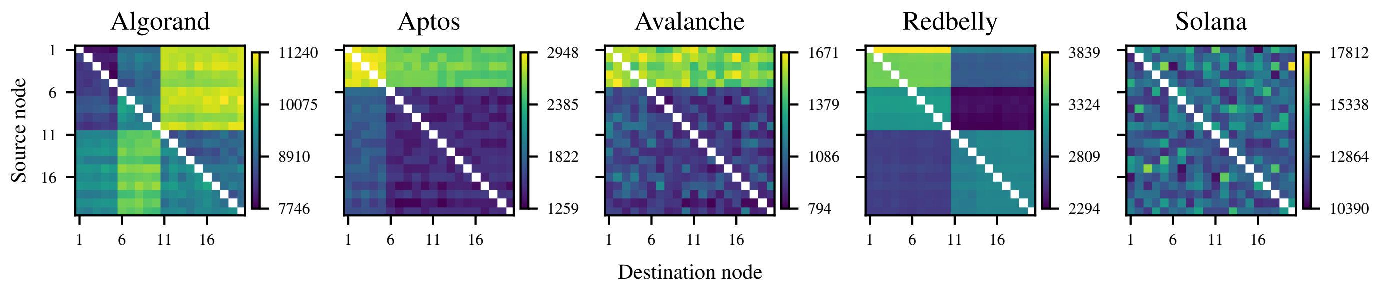

Figure 1: Heatmaps $M_{i,j}$ of the bandwidth used in KiB between sending node $n_i$ and receiving node $n_j$ for each blockchain, 20 nodes, 20 validators.

## 3 Experimental Settings

In this section, we explain how we deploy blockchain protocols and measure their bandwidth consumption.

### 3.1 Distributed system setup

Our experimental setup consists of a distributed system of 25 VMs running Ubuntu 24.04.1 LTS on top of a Proxmox cluster of physical servers, each equipped with 4x AMD Opteron 6378 16-core CPUs running at 2.40 GHz, 256 GB of RAM, and 10 GbE NICs. Each experiment runs a blockchain protocol with 5 client VMs and 20 blockchain node VMs. For the blockchain protocols, we used Algorand v3.27.0, Aptos v1.25.1, Avalanche C-Chain v1.12.1, Redbelly v0.36.2 and Solana Agave v2.0.20.

This 25-node setup is justified by recent studies [26] that confirmed that evaluations at small scale in controlled environments can accurately reproduce performance trends observed in geo-distributed settings. We define $N$ as the total number of blockchain nodes in the network, and $V$ as the size of the subset of validators participating in consensus such that $N, V \in \{5, 10, 15, 20\}$, $V \leq N$.

All experiments follow a fixed timeline: a 100 s "Setup" phase with no transactions, a 100 s "Workload" phase where transactions are sent, and a 100 s "Cooldown" phase with no transactions, allowing remaining transactions to commit. During the Workload phase, the 5 clients send transactions to the first 5 validators. The target load is distributed equally among clients (e.g., 40 TPS each for a total of 200 TPS). Each transaction is sent to a single node, which is then queried for finality using block streaming or polling.

The resources of each VM mimics intentionally the resources of a commodity computer run by an individual in a blockchain network. Note that this specification is lower than what some blockchains typically recommend, including Aptos [2], Avalanche [27] or Redbelly [30], however, strict hardware requirements on remote nodes remain hard to enforce and a unique configuration is necessary for our comparison.

### 3.2 Measuring peer-to-peer bandwidth

We implemented a fine-grained bandwidth monitoring system to capture traffic usage between blockchain nodes and clients. Our approach provides pairwise measurements of the traffic exchanged between each node in the network. The system utilizes STABL observer processes running on blockchain VMs and relies on the Linux `iptables` firewall infrastructure to perform non-intrusive packet accounting.

For each node under observation, we programmatically install a set of `iptables` rules. These rules create custom accounting chains that contain a specific rule for every other peer in the experiment. Each rule is configured to match packets based on their source (for incoming traffic) or destination (for outgoing traffic) IP address.

A monitoring script then periodically queries the byte counters associated with each of these per-peer rules and immediately resets them to zero. This process yields a time series where each data point represents the average transmission (TX) and reception (RX) rate over the preceding interval, allowing us to precisely analyze network behavior.

## 4 Varying Consumption

In order to illustrate how blockchains consume bandwidth, we present heatmaps with colors representing bandwidth usage. We then compare the transaction size produced by each blockchain.

### 4.1 Bandwidth consumptions as heatmaps

Figure 1 depicts one heatmap per blockchain protocol for nodes that all act as validators where the color of cell $M_{i,j}$ represents the bandwidth consumed by node $n_i$ when sending to node $n_j$. A warmer color (e.g., yellow) thus represents a higher bandwidth usage than a colder color (e.g., dark blue) while the white color indicates zero or negligible traffic. Note that nodes $n_1, \ldots, n_5$ are the nodes receiving transaction requests from the clients, while nodes $n_6, \ldots, n_{20}$ may receive messages from other nodes but not transactions directly from clients. With a maximum of 17,812 KiB, Solana consumes more bandwidth than the other tested blockchains, namely Algorand, Aptos, Avalanche and Redbelly. In particular, Algorand, the second most bandwidth consuming blockchain uses a maximum of 11,240 KiB. This 58% increase compared to Algorand can be due to several factors: more metadata sent per transaction or higher duplication of the same information.

The reason is probably that Solana exploits erasure coding in order to maximize dissemination of information despite failures [6]. Its lack of fault tolerance, already previously observed [20], probably motivated this design decision. Erasure coding relies on sending additional data in order to reconstruct the relevant information in case of partial loss. The other blockchains do not consume as much traffic likely because they do not use erasure coding.

### 4.2 Transaction sizes

In order to exclude other factors that could have invalidated our hypothesis that Solana uses more bandwidth due to erasure coding, we measured empirically the size of transactions sent by each blockchain. After all, it was previously noted that distributed ledger technologies like Corda send transactions as large as 8 KiB, which is an overkill [22].

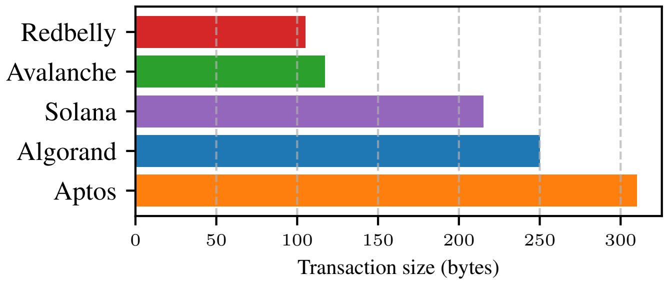

Figure 2: Transaction size per blockchain.

Fig. 2 compares the transaction sizes of the different blockchains. It shows in particular that, even though Solana uses more bandwidth as discussed above in Section 4, it does not use the largest transactions. Actually, Aptos' transaction size is 310 B compared to Solana's transaction size of 215 B, Aptos' transactions are 44% larger than Solana's. Finally, note that even Algorand's transactions, with a size of 250 B, are larger than Solana's transactions as well. As a result, the bandwidth usage of Solana is not a consequence of generating excessively large transactions.

## 5 Consumption Skewness over Routes

A high bandwidth usage does not necessarily induce a detrimental performance, especially when the bandwidth consumption is well-balanced over multiple routes. In fact, previous works have shown that balancing an amount of information that is quadratic in the number of network nodes over a quadratic number of routes of this network could be more efficient than reducing this amount of information to a linear factor but sent across the same routes [35]. It is thus important to understand how a blockchain protocol balances its bandwidth consumption over a network.

### 5.1 Some nodes exchange more data than others

In the Aptos heatmap of Fig. 1, we can see that some nodes of the Aptos network seem to consume more bandwidth than other nodes of the same network, and traffic is heavily unbalanced across nodes. Five nodes, the ones that receive the transactions sent by the clients, send more messages than all the other nodes as indicated by the top five rows in light colors. They also send more messages to themselves than to the rest of the nodes, as indicated by the yellow 5-by-5 sub-matrix of the top left corner of the heatmap. This is due to its Quorum Store optimization [7] that puts more load on the receivers of transactions than on the rest of the network by requiring them to collect signatures. More precisely, these validator nodes have to sign transaction batch digests and collect the produced signatures from a quorum of blockchain nodes to form a "proof-of-store". This signature collection puts inevitably more bandwidth pressure on the nodes responsible for signing.

In the Solana heatmap of Fig. 1, the color shows that there is no clear per-node distinction as no column or row stands out. As a result, the bandwidth consumption of Solana appears generally more balanced than the one of Aptos. It is interesting to note that Solana, which consumes a lot of bandwidth as we showed in Section 4, manages to balance the load pretty well.

Finally, the Algorand heatmap of Fig. 1 shows some imbalance in that the second half of the nodes $n_{11}, \ldots, n_{20}$ receive and send more messages than others.

### 5.2 Some nodes send more than they receive in Avalanche and Redbelly

Another interesting dimension to consider is the level of unbalance between sending and receiving traffic. The nodes receiving transactions could either consume more bandwidth by propagating the information or, instead, they could consume less than the nodes agreeing on which block to append. For example, the Avalanche heatmap of Fig. 1 shows that the nodes of Avalanche that receive transactions generate more traffic than what they receive.

The Avalanche heatmap of Fig. 1 shows the five top rows in lighter color than the five left columns. This indicates that the five Avalanche nodes that receive transactions send more data than they receive. By contrast with Aptos, they do not need to send more data to themselves than to the rest of the network. This is explained by the fact that they have to propagate the transactions they receive to the rest of the network without needing to collect signatures from each other.

The Redbelly heatmap of Fig. 1 shows one particular node sending more messages than others. This is explained by having a weak coordinator that sends a particular message in the first round of the DBFT binary consensus protocol [11]. Note that this coordinator is weaker than a leader in that the consensus terminates even when it is faulty. Generally, the first five nodes send more messages than others because they forward the transactions that this experiment sends them. Finally, the remaining differences could be due to the node placement and the reordering of messages, requiring some nodes to request the batch of proposed transactions from other nodes.

### 5.3 The dissemination reliability of Solana requires more traffic

Bandwidth consumption can have some advantages, for example, when redundant information copes with packet losses. The trade-off between bandwidth efficiency and dissemination reliability is particularly apparent in Solana. Previous research has highlighted Solana's ability to tolerate packet losses better than other blockchains [20]. Our experiments quantify the cost of this dissemination reliability.

The Solana heatmap of Fig. 1 shows the bandwidth consumption is well balanced and very high. Unlike other protocols that attempt to minimize redundant transmissions (resulting in the dark blue or empty regions seen in Algorand or Redbelly), Solana appears to utilize a "flood" or highly redundant propagation mechanism, likely related to its Turbine block propagation protocol and erasure coding schemes. While this results in significantly higher total bandwidth consumption, it ensures that data is recoverable and available to all nodes, even in the presence of network failures.

## 6 Consumption Skewness over Time

In order to better understand the bandwidth usage, we measured the bandwidth consumed at the three different stages of our experiments (Setup, Submission, and Cooldown) as defined in Section 3.

Fig. 3 differentiates the Transmitted (TX) data, which are sent from nodes to clients, from the Received (RX) data, which are sent from clients to nodes, when 5 clients submit 19,995 transactions (3,999 each) to 5 nodes.

### 6.1 Impact of communication protocols on idle traffic

A distinct disparity is visible in the overhead traffic during the non-submission phases (Setup and Cooldown).

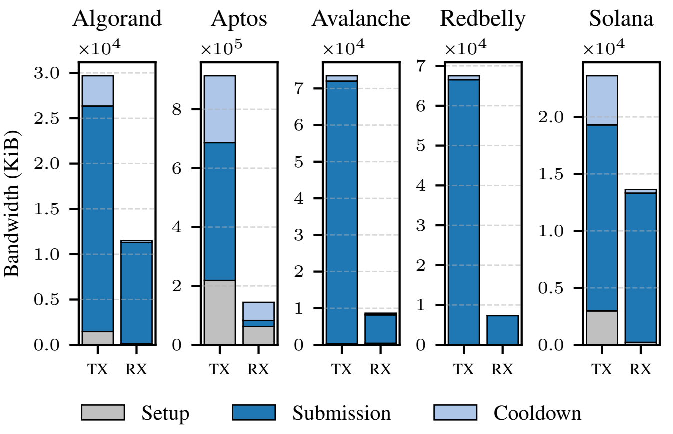

Figure 3: Client-side network traffic (TX and RX) stacked by experimental phase. Y-axis scales differ across subplots to accommodate variations in magnitude.

Aptos exhibits significantly higher bandwidth consumption during these idle periods compared to other blockchains. For instance, during Phase 0 (Setup), Aptos nodes transmitted 218,856 KiB of data, whereas Avalanche and Redbelly transmitted only 298 KiB and 11 KiB, respectively.

This massive overhead in Aptos is attributed to its client communication architecture. While Avalanche, Redbelly, and Solana utilize WebSocket streaming to push updates to clients efficiently, Aptos clients rely on polling. Consequently, Aptos clients repeatedly request data even when blocks are empty, resulting in substantial bandwidth usage (≈227,000 KiB in Phase 2) despite the absence of new transaction submissions.

### 6.2 Block verification and data efficiency

During the Submission phase (Phase 1), a clear divergence in data efficiency emerges between Solana and the chains compatible with the Ethereum Web3 WebSocket API (Avalanche, Redbelly) that use the same methods to send and listen to blocks as Ethereum.

**Full block transmission** These protocols exhibit a high ratio of TX to RX traffic. For example, Redbelly received 7,314 KiB of transaction data (RX) but transmitted 66,511 KiB (TX) back to the clients. This amplification occurs because these protocols require the node to broadcast the entire block (containing transaction bodies and metadata) to every client for verification. Even though a client only needs transaction hashes to confirm their finality, it must download the full block payload.

**Signature-based verification** Solana demonstrates a more balanced traffic profile during submission (RX: 13,099 KiB vs. TX: 16,325 KiB). Despite Solana having a relatively large raw transaction size (215 B) compared to Redbelly (105 B) or Avalanche (117 B), its outgoing traffic is significantly lower. This efficiency stems from Solana's

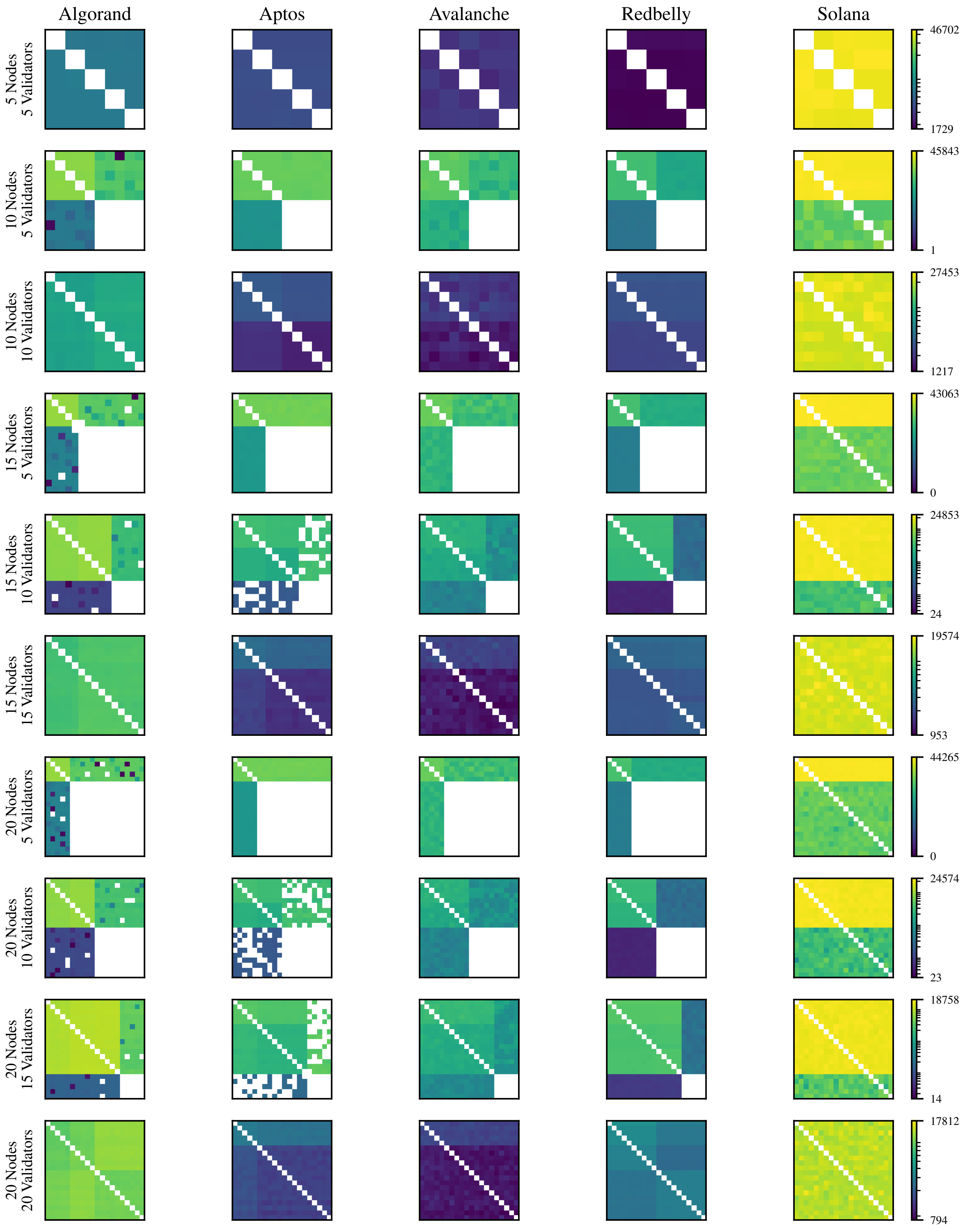

Figure 4: Heatmaps $M_{i,j}$ of the bandwidth used in KiB between sending node $n_i$ and receiving node $n_j$ for each blockchain.

verification mechanism; clients requesting block confirmation do not need to download the full block body. Instead, they request only the hashes or signatures of committed transactions and compare them against their locally stored payloads. This selective data retrieval significantly reduces the egress bandwidth required by the nodes.

In summary, while the raw transaction sizes (ranging from 105 B on Redbelly to 310 B on Aptos, as displayed in Fig. 2) play a role in bandwidth usage, the dominant factors influencing network traffic are the choice of transport protocol (polling vs. WebSockets) and the block verification strategy (full block download vs. hash comparison).

# 7 Consumption Skewness over Scale

In this section, we analyze the scalability of the bandwidth consumption, denoted as $S$, as a function of the number $N$ of nodes and the number $V$ of validators across the different blockchains. Interestingly, we identify that the bandwidth of Aptos, Avalanche and Solana increases with $N$ and the bandwidth of Algorand, Aptos, Avalanche and Redbelly increases with $V$.

## 7.1 Large consumption variations depending on network scale

Fig. 4 depicts a comprehensive grid of heatmaps for every blockchain across different combinations of number $N$ of nodes and number $V$ of validators. In each heatmap, the bandwidth consumed by node $n_i$ sending to node $n_j$ is represented by the color in cell $M_{i,j}$.

The raw data reveals significant disparities in the magnitude of data exchange across the different protocols. In a fully interconnected small network (5 nodes, all validators), the difference in per-link bandwidth is striking: while Redbelly and Avalanche maintain a lean footprint with approximately 1,700 KiB and 2,800 KiB transmitted per pair respectively, Algorand consumes considerably more, averaging around 6,500 KiB per link. Solana, however, operates at a completely different order of magnitude, with cells in the 5-node configuration consistently showing over 42,000 KiB of traffic—nearly 25 times the bandwidth usage of Redbelly for the same workload.

Furthermore, the data highlights a clear hierarchy of network load based on node roles. In mixed configurations (e.g., 20 nodes with 5 validators), the bandwidth intensity within the validator group (the top-left $5 \times 5$ sub-matrix) is drastically higher than the traffic involving non-validator nodes. For instance, in the 10-5 setup, for Aptos and Algorand, the traffic between two validators remains in the thousands of KiB (e.g., $\approx$3,600 KiB for Aptos), whereas traffic originating from non-validator nodes often drops to the low hundreds (e.g., $\approx$200 KiB), illustrating a highly centralized bandwidth burden on the consensus committee.

## 7.2 Validators communicate more with themselves

A distinct segmentation of the network topology is visible in most protocols. For Algorand, Aptos, Avalanche, and Redbelly, the top-left $V \times V$ sub-matrix is consistently dense and bright, confirming that validators communicate heavily among themselves to achieve consensus. However, the behavior regarding non-validator nodes varies significantly.

As observed, Solana is the unique outlier. It is the only blockchain where the non-validator nodes communicate directly with other non-validator nodes (the bottom-right quadrant of the heatmaps). For instance, in the 10-5 configuration, the heatmap is uniformly populated, indicating a full mesh topology where every node, regardless of its role, exchanges data with every other node.

In contrast, blockchains like Redbelly and Avalanche show a clear separation. While validators send data to non-validators (top-right quadrant) and receive data from them (bottom-left quadrant), non-validator nodes do not communicate with each other. This is evident in the 10-5 matrices for Avalanche and Redbelly, where the bottom-right $5 \times 5$ sub-matrix is largely empty.

Finally, Algorand and Aptos exhibit the strictest separation in certain configurations. Non-validator nodes act almost exclusively as passive receivers or pull-based clients. In Algorand's 10-5 configuration, while validators send data to non-validators (rows 0-4 to columns 5-9), the traffic from non-validators back to validators is negligible, and traffic between non-validators is nonexistent.

## 7.3 Aptos pattern change with the number of validators

Fig. 4 reveals a dynamic dissemination strategy in Aptos that depends on the ratio of validators to the total network size. In configurations where the number of validators is small relative to the network size, validators appear to broadcast to all non-validators. For example, in the 10-5 configuration and 20-5 configuration, the top-right quadrants are dense, showing that the 5 validators are sending data to all 5 (or 15) non-validator nodes.

However, as the number of validators increases, Aptos shifts strategy to reduce bandwidth overhead. In the 15-node, 10-validator configuration, the top-right quadrant becomes sparse. Specific validators only communicate with specific non-validator nodes rather than broadcasting to the entire set. This suggests a sharding or randomized gossip approach to dissemination when the validator set is large, likely to prevent bandwidth saturation on individual nodes.

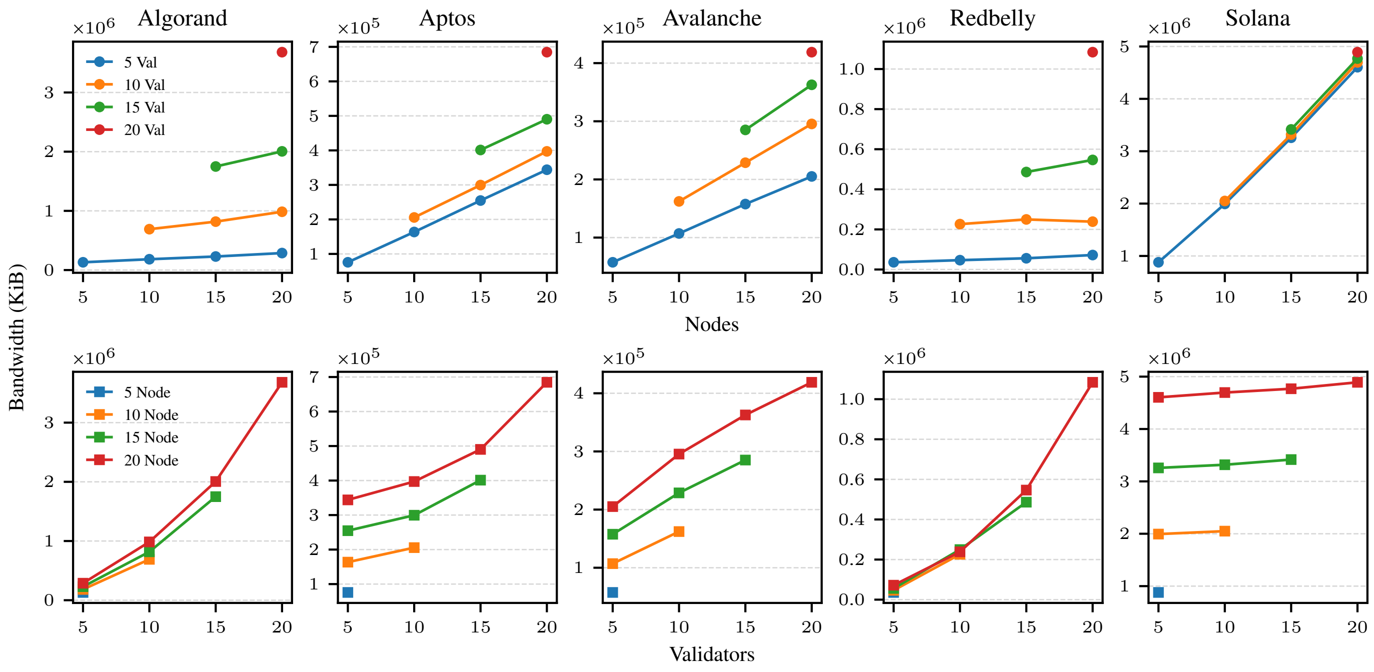

Figure 5: Total bandwidth consumption (KiB) for blockchains under test. The top row plots bandwidth against the number of nodes (grouped by validator count), while the bottom row plots bandwidth against the number of validators (grouped by node count).

## 7.4 Blockchains vary with regards to bandwidth scalability

Fig. 5 depicts the trends of bandwidth consumption depending on the scale, where each figure provides: line plots showing $S$ as a function of the total number $N$ of nodes, and line plots showing $S$ as a function of the number $V$ of validators. Here, $S$ represents the total sum of bandwidth measured in KiB during the experiment duration.

We can clearly see from Fig. 5 that $S_{Solana}$ bandwidth consumption increases primarily with the number of nodes, regardless of the number of validators. For instance, with 5 validators, increasing the total node count from 5 to 20 causes bandwidth to surge from 879,268 KiB to 4,604,375 KiB—a five-fold increase. However, keeping the node count fixed at 20 and increasing validators from 5 to 20 results in a negligible increase from 4,604,375 KiB to 4,891,524 KiB. This confirms that Solana requires all nodes to communicate with each other, creating a high-bandwidth mesh topology dependent on the network size rather than the validator set size.

In contrast, blockchains $b \in \{Algorand, Redbelly\}$ the bandwidth $S_b$ depend almost entirely on the number of validators. Adding non-validator nodes adds very little overhead. For Algorand, with 5 validators, increasing nodes from 10 to 20 only increases usage from 182,790 KiB to 287,032 KiB. However, increasing validators has a massive impact: with 20 nodes, moving from 5 to 20 validators causes usage to skyrocket from 287,032 KiB to 3,681,191 KiB—an increase of over 12 times. Redbelly exhibits an even more drastic ratio, jumping from 72,098 KiB (20 nodes, 5 validators) to 1,084,185 KiB (20 nodes, 20 validators).

Finally, in two blockchains, Aptos and Avalanche, the bandwidth consumption grows with both the number of validators and the number of nodes. Note that the bandwidth increase appears superlinear with the number of validators in Aptos but sublinear in Avalanche. In Aptos, with 20 nodes, the bandwidth grows moderately between 5 and 15 validators (343,968 KiB to 490,237 KiB) but jumps sharply when reaching 20 validators (684,649 KiB). Conversely, Avalanche shows diminishing returns in bandwidth growth as validators are added to a 20-node network, rising from 205,154 KiB (5 validators) to 418,852 KiB (20 validators), indicating a more consistent propagation overhead.

Based on these trends, we can classify blockchains in three categories:

1. **Node-dependent (Solana)** Bandwidth consumption increases with the total network size ($N$), indicating a topology where all nodes propagate data heavily.

2. **Validator-dependent (Algorand and Redbelly)** Bandwidth consumption increases primarily with the size of the consensus committee ($V$). Non-validator nodes are passive consumers.

3. **Hybrid (Avalanche and Aptos)** Bandwidth consumption increases with both $N$ and $V$, suggesting a topology where non-validator nodes participate in propagation.

## 8 Consumption Variation with TPS

The bandwidth consumption could simply be a consequence of the rate at which our experiments send transactions. Below, we vary this sending rate to observe how it impacts bandwidth consumption. To this end, we conducted a set of experiments where the total workload size was kept constant at approximately 20,000 transactions. We varied the target TPS from 100 to 500, inversely adjusting the experiment duration (from 200 s down to 40 s) to maintain the fixed transaction count (TPS $\times$ Duration $\approx$ Constant). Fig. 6 presents these results from the perspective of total absolute bandwidth.

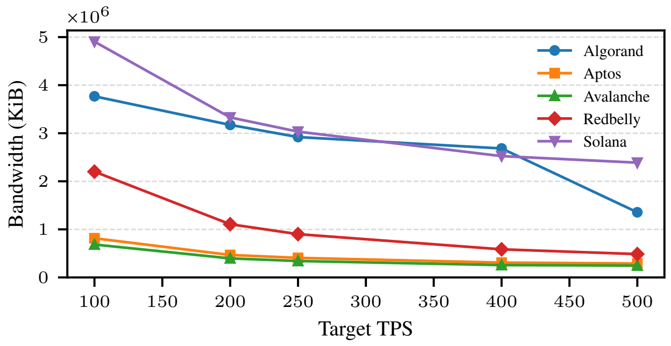

Figure 6: Total bandwidth consumption (KiB) vs. Target TPS for blockchains under test under varying load rates. The workload was fixed at 20,000 total transactions per run, with experiment duration scaling inversely to TPS (200 s down to 40 s).

Fig. 6 plots the total absolute bandwidth consumed by each blockchain. A consistent trend is visible across all protocols: total bandwidth consumption decreases as TPS increases. Since the number of transactions is fixed, this decrease indicates that a significant portion of bandwidth usage is time-dependent rather than transaction-dependent. At lower TPS (longer duration), "background" traffic, such as heartbeats, empty block proposals, and consensus maintenance, accumulates, inflating the total footprint.

Solana consistently consumes the highest absolute bandwidth, ranging from approximately 4.7 GiB (4,905,820 KiB) at 100 TPS down to 2.3 GiB (2,387,814 KiB) at 500 TPS. Algorand follows as the second most bandwidth-intensive chain (3.6 GiB to 1.3 GiB), while Avalanche and Aptos remain the most efficient, with Aptos consuming as little as 283 MiB (289,834 KiB) at 500 TPS. Notably, in terms of absolute bandwidth, Algorand consistently consumes more than Redbelly across all data points (e.g., at 100 TPS: Algorand $\approx$3.6 GiB vs. Redbelly $\approx$2.1 GiB).

## 9 Related Work

**Blockchain benchmarking** Existing frameworks like Blockbench [17] and Hyperledger Caliper [24] standardize the evaluation of throughput, latency, and fault tolerance. Gromit [28] further addresses ad-hoc testing limitations. However, these frameworks prioritize execution capacity (TPS) and treat the network layer as secondary. Our work builds upon Diablo [19] and STABL [20] to rigorously isolate bandwidth consumption, a critical metric overlooked by execution-focused benchmarks.

**Network traffic and block propagation** Early studies analyzed propagation delays in Bitcoin [14], leading to bandwidth optimization protocols like Graphene [29], Compact Blocks [10], and the BloXroute [25] Layer-0 CDN. While these works propose techniques to minimize traffic, our paper provides a comparative measurement of modern Layer-1 protocols. We reveal that contemporary systems like Solana often prioritize data redundancy over the bandwidth efficiency emphasized in earlier Bitcoin and Ethereum research.

**Communication complexity vs. empirical reality** While theoretical literature favors linear communication complexity ($O(n)$) protocols like HotStuff [38] over quadratic BFT implementations [5], theoretical bounds often fail to predict real-world behavior. By contrast, Voron and Gramoli [35] showed empirically that quadratic complexity can achieve significantly better performance when well balanced across a quadratic number of routes of a WAN. Di Perna et al. [15] showed that dense network topologies improve performance under load, a finding correlated with higher energy usage [16]. Our empirical results confirm this divergence between redundancy-heavy architectures like Solana and the lean traffic profiles of Redbelly.

## 10 Conclusion

This paper compares the bandwidth consumption of blockchains for the first time. To this end, we measured empirically the traffic of five major layer-1 blockchains Algorand, Aptos, Avalanche, Redbelly, and Solana.

Our results show that transport protocols (polling vs. WebSockets) and block propagation strategies impact bandwidth more than transaction size. Solana consumes up to 58% more bandwidth than Algorand due to redundancy, while Aptos suffers from high idle overhead. Crucially, Solana scales with network size, whereas Algorand and Redbelly scale with validator count.

## Acknowledgment

This research is supported under Australian Research Council Discovery Project funding scheme (project number 250101739) entitled "Fair Ordering of Decentralised Access to Resources".

## References

[1] Anza. Turbine block propagation. Accessed: Mar. 7, 2025. URL: `https://docs.anza.xyz/consensus/turbine-block-propagation`.

[2] Aptos. Node requirements, February 2025. Accessed: Mar. 7, 2025. URL: `https://aptos.dev/en/network/nodes/validator-node/node-requirements`.

[3] Aptos Foundation. The Aptos blockchain: Safe, scalable, and upgradeable Web3 infrastructure, August 2022. URL: `https://aptosfoundation.org/whitepaper/aptos-whitepaper_en.pdf`.

[4] Avalanche. AvalancheGo configs and flags, 2024. Accessed: Aug. 31, 2024. URL: `https://docs.avax.network/nodes/configure/configs-flags`.

[5] Miguel Castro and Barbara Liskov. Practical Byzantine fault tolerance. In *Proceedings of the Third Symposium on Operating Systems Design and Implementation*, OSDI '99, pages 173–186, USA, 1999. USENIX Assoc.

[6] Ryan Chern. Turbine: Block propagation on Solana, October 2023. Accessed: Mar. 7, 2025. URL: `https://www.helius.dev/blog/turbine-block-propagation-on-solana`.

[7] Brian Cho. AIP-26 – Quorum Store, 2023. Accessed: Jan. 9, 2026. URL: `https://github.com/aptos-foundation/AIPs/blob/main/aips/aip-26.md`.

[8] Brian Cho and Alexander Spiegelman. Quorum Store: How consensus horizontally scales on the Aptos blockchain, May 2023. Accessed: Mar. 7, 2025. URL: `https://medium.com/aptoslabs/quorum-store-how-consensus-horizontally-scales-on-the-aptos-blockchain-988866f6d5b0`.

[9] Mauro Conti, Ankit Gangwal, and Michele Todero. Blockchain trilemma solver Algorand has dilemma over undecidable messages. In *Proceedings of the 14th International Conference on Availability, Reliability and Security*, New York, NY, USA, 2019. ACM.

[10] Matt Corallo. Compact block relay. bip 152, 2016. Accessed: Jan. 12, 2026. URL: `https://github.com/bitcoin/bips/blob/master/bip-0152.mediawiki`.

[11] Tyler Crain, Vincent Gramoli, Mikel Larrea, and Michel Raynal. DBFT: efficient leaderless Byzantine consensus and its application to blockchains. In *17th IEEE International Symposium on Network Computing and Applications NCA*, pages 1–8. IEEE, 2018.

[12] Tyler Crain, Christopher Natoli, and Vincent Gramoli. Red Belly: a secure, fair and scalable open blockchain. In *Proceedings of the 42nd IEEE Symposium on Security and Privacy (S&P)*. IEEE, May 2021.

[13] George Danezis, Lefteris Kokoris-Kogias, Alberto Sonnino, and Alexander Spiegelman. Narwhal and Tusk: a DAG-based mempool and efficient BFT consensus. In *Proceedings of the Seventeenth European Conference on Computer Systems*, pages 34–50, Rennes France, March 2022. ACM.

[14] Christian Decker and Roger Wattenhofer. Information propagation in the bitcoin network. In *IEEE P2P 2013 Proceedings*, pages 1–10, 2013.

[15] Vincenzo Di Perna, Marco Bernardo, Francesco Fabris, Sebastiao Amaro, Miguel Matos, and Valerio Schiavoni. Impact of network topologies on blockchain performance. In *Proc. 19th ACM International Conference on Distributed and Event-Based Systems (DEBS)*, June 2025.

[16] Vincenzo P. Di Perna, Valerio Schiavoni, Francesco Fabris, and Marco Bernardo. Blockchain energy consumption: Unveiling the impact of network topologies. In *2025 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pages 1–10, 2025.

[17] Tien Tuan Anh Dinh, Ji Wang, Gang Chen, Rui Liu, Beng Chin Ooi, and Kian-Lee Tan. Blockbench: A framework for analyzing private blockchains. In *Proceedings of the 2017 ACM International Conference on Management of Data*, page 1085–1100, 2017.

[18] Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos, and Nickolai Zeldovich. Algorand: Scaling Byzantine agreements for cryptocurrencies. In *Proceedings of the 26th Symposium on Operating Systems Principles (SOSP)*, pages 51–68. ACM, 2017.

[19] Vincent Gramoli, Rachid Guerraoui, Andrei Lebedev, Chris Natoli, and Gauthier Voron. Diablo: A benchmark suite for blockchains. In Giuseppe Antonio Di Luna, Leonardo Querzoni, Alexandra Fedorova, and Dushyanth Narayanan, editors, *Proceedings of the Eighteenth European Conference on Computer Systems (EuroSys)*, pages 540–556. ACM, 2023.

[20] Vincent Gramoli, Rachid Guerraoui, Andrei Lebedev, and Gauthier Voron. Stabl: The sensitivity of blockchains to failures. In *Proceedings of the 26th International Middleware Conference*, Middleware '25, page 202–214, New York, NY, USA, 2025. Association for Computing Machinery.

[21] Hivelocity. The ultimate guide to solana validator infrastructure, 2026. Accessed: Jan. 12, 2026. URL: `https://www.hivelocity.net/kb/solana-validator-infrastructure/`.

[22] David Hyland, João Sousa, Gauthier Voron, Alysson Bessani, and Vincent Gramoli. Ten myths about blockchain consensus. In Sushmita Ruj, Salil S. Kanhere, and Mauro Conti, editors, *Blockchains*, volume

105, pages 3–24. Springer International Publishing, Cham, 2024.

[23] IETF. RFC 9000 – QUIC: A UDP-based multiplexed and secure transport, 2021.

[24] Dave Kelsey. Hyperledger Caliper, 2024. Accessed: Jan. 12, 2026. URL: `https://hyperledger.github.io/caliper/`.

[25] Uri Klarman, Soumya Basu, Aleksandar Kuzmanovic, and Emin Gün Sirer. bloxroute: A scalable trustless blockchain distribution network, 2019. Accessed: Jan. 12, 2026. URL: `https://bloxroute.com/wp-content/uploads/2019/11/bloXrouteWhitepaper.pdf`.

[26] Andrei Lebedev and Vincent Gramoli. On the relevance of blockchain evaluations on bare metal. In *7th Symposium on Distributed Ledger Technologies (SDLT)*. Springer Nature Singapore, 2023.

[27] Michael Nadeau. The fundamentals of Avalanche, November 2023. Accessed: Mar. 7, 2025. URL: `https://tokenterminal.com/crypto-research/avalanche#decentralization`.

[28] Bulat Nasrulin, Martijn De Vos, Georgy Ishmaev, and Johan Pouwelse. Gromit: Benchmarking the performance and scalability of blockchain systems. In *2022 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS)*, pages 56–63, Newark, CA, USA, August 2022. IEEE.

[29] A. Pinar Ozisik, Gavin Andresen, Brian N. Levine, Darren Tapp, George Bissias, and Sunny Katkuri. Graphene: efficient interactive set reconciliation applied to blockchain propagation. In *Proceedings of the ACM Special Interest Group on Data Communication*, SIGCOMM '19, page 303–317, New York, NY, USA, 2019. Association for Computing Machinery.

[30] Redbelly. Node hardware specification requirements — Vine (Redbelly developer portal). Accessed: Mar. 7, 2025. URL: `https://vine.redbelly.network/nds-node-hardware-specification-requirements`.

[31] Team Rocket, Maofan Yin, Kevin Sekniqi, Robbert van Renesse, and Emin Gün Sirer. Scalable and probabilistic leaderless BFT consensus through metastability. Technical report, arXiv, August 2020. URL: `http://arxiv.org/abs/1906.08936`.

[32] Deepal Tennakoon and Vincent Gramoli. Deconstructing the Smart Redbelly blockchain. *IEEE Trans. Comput.*, 2024.

[33] Deepal Tennakoon, Yiding Hua, and Vincent Gramoli. Smart Redbelly blockchain: Reducing congestion for Web3. In *2023 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 940–950, St. Petersburg, FL, USA, May 2023. IEEE.

[34] Gauthier Voron and Vincent Gramoli. Dispel: Byzantine SMR with distributed pipelining. Technical Report 1912.10367, arXiv, 2020.

[35] Gauthier Voron and Vincent Gramoli. Invited paper: Planetary scale byzantine consensus. In *Proceedings of the 5th Workshop on Advanced Tools, Programming Languages, and PLatforms for Implementing and Evaluating Algorithms for Distributed Systems*, ApPLIED 2023, New York, NY, USA, 2023. Association for Computing Machinery.

[36] Timo Wevelsiep. Public internet egress costs: Aws vs azure vs gcp vs hetzner comparison, 2025. Accessed: Jan. 12, 2026. URL: `https://wz-it.com/en/blog/public-internet-egress-costs-comparison-aws-azure-gcp-hetzner/`.

[37] Anatoly Yakovenko. Solana: A new architecture for a high performance blockchain v0.8.13, 2021. Accessed: Jan. 12, 2026. URL: `https://solana.com/solana-whitepaper.pdf`.

[38] Maofan Yin, Dahlia Malkhi, Michael K. Reiter, Guy Golan Gueta, and Ittai Abraham. HotStuff: BFT consensus with linearity and responsiveness. In *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing*, 2019.