IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 17 January 2024, accepted 5 February 2024, date of publication 8 February 2024, date of current version 15 February 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3363833

APPLIED RESEARCH

Validating the Blockchain Benchmarking Framework Through Controlled Deployments of XRPL and Ethereum

MARIOS TOULOUPOU, KLITOS CHRISTODOULOU, AND MARINOS THEMISTOCLEOUS
Department of Digital Innovation, University of Nicosia, Nicosia 2417, Cyprus

Corresponding author: Marios Touloupou (touloupos.m@unic.ac.cy)

This work was supported in part by the University Blockchain Research Initiative (UBRI) Project, funded by the Ripple's Impact Fund, a fund of the Silicon Valley Community Foundation, under Grant 2021-244121.

ABSTRACT In the evolving domain of blockchain, a critical challenge lies in the performance analysis of blockchains under controlled test conditions. This paper focuses on validating the Blockchain Benchmarking Framework (BBF), developed for the evaluation of blockchain protocols in a controlled environment. The BBF's robustness and versatility are demonstrated through its application to the official Docker clients of Ripple's XRP Ledger (XRPL) and Ethereum, deployed in private, local and controlled environments. These deployments are utilized to simulate network dynamics, transaction throughput, and resilience in a variety of scenarios. Our methodology encompasses tests ranging from standard operational conditions to adverse scenarios, including node failures and simulated double-spend attacks. These controlled environments are essential for evaluating the BBF's efficacy in stress testing blockchain protocols and assessing their stability and robustness. The BBF's ability to accurately capture and analyze performance characteristics is highlighted, providing insights into the operational mechanics, scalability, and resilience of these blockchain clients. The findings emphasize the BBF's adaptability and effectiveness in managing different blockchain protocols, reaffirming its potential for broader application in pre-launch testing and analysis of blockchain performance. This study contributes to the understanding of how blockchain clients can be preliminarily assessed before mainnet deployment as well as to validate all the design decisions made by the protocol under different settings and synthetic scenarios.

INDEX TERMS Benchmarking framework, blockchain applications, blockchain resilience, blockchain technology, performance analysis.

I. INTRODUCTION

Blockchain technology, a groundbreaking innovation in digital transaction systems, has profoundly transformed the landscape of digital economies across the globe. With a robust capacity for secure data management and transaction processing, the technology has found applications across diverse industries, including finance, healthcare, supply chain, and public governance [1]. The potential for a decentralized and irreversible digital ledger to fundamentally transform conventional company processes has generated considerable attention and substantial investment from industry pioneers, practitioners, and academics [2]. However, a critical issue is the emergence of a multitude of blockchains platforms, each presenting unique features, functionalities, and underlying design philosophies. These platforms, such as Bitcoin, Ethereum, XRPL and Hyperledger, have sparked extensive debates and comparative analyses to unravel their relative strengths, weaknesses, and best-fit application scenarios [3]. Such a wide spectrum of options makes the choice of an appropriate blockchain platform a complex task, compounded by the absence of a universally accepted

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

standard or framework to compare their performance under diverse operating conditions [4].

A major challenge in benchmarking blockchain platforms arises from the heterogeneity of the technology landscape. Developers and decision-makers are often tasked with making informed decisions, requiring a deep understanding of the trade-offs and performance metrics of each platform [5]. The dynamics of each blockchain protocol, encompassing aspects such as consensus algorithms (CA), transaction validation processes, security features, and scalability solutions, present byzantine variables that influence performance outcomes [6]. Furthermore, the intricate nature of these technologies underscores the imperative need for rigorous pre-launch testing of blockchain protocols. Such testing is crucial for identifying and mitigating potential issues that could compromise the security, functionality, and scalability of blockchain applications before their deployment in live environments. The lack of universal benchmarks and evaluation frameworks accentuates these complexities, making it challenging to assess the platforms' robustness and resilience under diverse workloads and operating conditions [7]. In this context, our proposed Blockchain Benchmarking Framework (BBF) aims to address this critical gap, providing a pragmatic and flexible tool that enables an unbiased and thorough comparison of blockchain platforms. The BBF's design caters specifically to the pre-launch testing phase, offering developers and researchers a means to simulate real-world conditions and assess the impact of design choices in a controlled, isolated setting.

Building on our previous work where we introduced the BBF [8], this study examines further into its practical application. The primary aim of this study is to rigorously evaluate the BBF in the context of two distinct blockchain protocols: Ethereum, operating under a Proof of Authority (PoA) CA, and the XRP Ledger (XRPL), which uses a Byzantine Fault Tolerance (BFT)-like CA known as the Ripple Protocol Consensus Algorithm (RPCA). The evaluation encompasses a comprehensive analysis of essential performance metrics such as latency, throughput, consensus time, and security aspects, including the protocols' resilience to double-spend attacks and node failure or crash scenarios. Most importantly, this study demonstrates the potential of the BBF as a tool for methodically assessing and validating the technical specifications and design choices of blockchain platforms. As a result, it provides significant knowledge regarding the operational intricacies and viability of various blockchain protocols, considering the viewpoints of both clients and ecosystems.

One of the standout features of the BBF is its technical depth and adaptability. Unlike conventional benchmarking tools, the BBF employs a flexible benchmark model that can be tailored to match the unique characteristics of different blockchain architectures. These include, but are not limited to, variations in permissioning (public and private), trust models (permissioned and permissionless), and data models (UTXO-based and account-based platforms) [9]. This flexibility enables the BBF to provide a comprehensive benchmark suite that can address unique platform features, application requirements, and evaluate their impact on performance and scalability.

To validate the efficacy of the BBF and to provide a practical demonstration of its benchmarking capabilities, we selected two diverse blockchain platforms - Ripple's XRPL and Ethereum. Our choice of XRPL and Ethereum was guided by their contrasting algorithmic and architectural design, which are indicative of the broad spectrum of existing blockchain platforms. While XRPL is optimized for high-speed, high-volume financial transactions, Ethereum is characterized by its versatile, Turing-complete smart contract capabilities [10]. It is essential to note that in our study, the BBF was applied to the official Docker clients of these platforms, providing a realistic testing environment for our analysis. These fundamental differences offer a rich context for demonstrating the BBF's robustness and versatility in handling heterogeneous platforms and providing insightful comparisons.

Through an examination of the performance and robustness of XRPL and Ethereum under a range of scenarios, our objective is to shed light on the process of selecting a platform that is well-informed and tailored to the performance needs and application specifications. Moreover, our objective is to enhance the value proposition of BBF as an all-encompassing and essential tool for forthcoming investigations, assessments, and implementations of blockchain technology.

The rest of this paper is structured as follows: Section II provides an overview on the background of this study as well as brief literature review. Sections III describes the methodology followed towards the evaluation of the BBF while section IV demonstrates the two use cases of XRPL and Ethereum. Section V synthesizes the findings from the two use cases, offering an analysis that highlights their relative strengths, weaknesses, performance under various conditions, and suitability for different applications. Finally, Section VI concludes the paper with a broad-ranging discussion on the potential impact, practical usefulness, limitations, and potential improvements to current blockchain technologies based on our research findings.

II. BACKGROUND AND LITERATURE REVIEW

Blockchain technology has increasingly gained attention due to its potential to disrupt traditional systems and processes. Its decentralization, security, transparency, and data integrity features position it as an innovative solution in various industries [11]. Despite the rapid adoption and development of blockchain technology, particularly the Ethereum and XRPL platforms, there exists a significant gap in literature regarding systematic performance analysis and comparison of these platforms. This observation aligns with the findings from our previous research [8], where we employed the multi-vocal Systematic Literature Review (SLR) approach, as outlined in [12], to further explore the existing body of literature.

A. BLOCKCHAIN TECHNOLOGY

Blockchain is a distributed, decentralized, and immutable ledger technology initially designed to support cryptocurrencies like Bitcoin [13]. At the core of blockchain technology are blocks of transactions linked together in a chain. Transactions are validated and added to the block by network nodes through a CA, varying depending on the blockchain variant – Proof of Work (PoW), Proof of Stake (PoS), etc. [14]. This technology has been extended beyond cryptocurrency into a variety of applications due to its inherent security and transparency.

B. THE BLOCKCHAIN BENCHMARKING FRAMEWORK

The BBF [8], [15], [16], as illustrated in **Fig. 1**, represents a solution designed for the evaluation of blockchain protocols. At the forefront of its design is a three-layer architecture, ensuring thorough and efficient performance analysis. The genesis of the BBF involved the development of an array of metrics and testing methodologies, essential for a comprehensive assessment of various blockchain protocols.

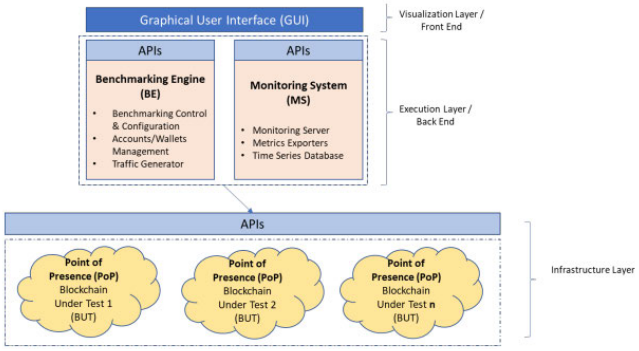

FIGURE 1. The blockchain benchmarking framework.

The core of the BBF consists of a Visual Analytics Layer, featuring a user-friendly web interface that promotes ease of use, especially for non-technical users. This layer is complemented by an Execution Layer, which includes a Benchmarking Engine (BE) for the automated deployment and monitoring of blockchain protocols. The BE was specifically utilized for the deployment and analysis of the official Docker clients of Ripple's XRP Ledger (XRPL) and Ethereum in our experiments. The BE adopts a microservice approach, enhancing scalability, and incorporates essential components such as Control & Configuration, Accounts/Wallets Management, and a Traffic Generator.

In addition, the Infrastructure Layer facilitates the deployment and stress-testing of blockchain protocols, underpinning the framework's overall functionality. The BBF's architecture is further bolstered by a Monitoring System (MS) that integrates tools like Prometheus[1] and Grafana,[2] providing robust capabilities in data collection, storage, and visualization.

[1]https://prometheus.io/
[2]https://grafana.com/

C. ETHEREUM

Ethereum, proposed by Buterin in 2013, was the first to introduce smart contracts, allowing developers to build and deploy decentralized applications (DApps) on its platform [17]. Ethereum, originally operating on the PoW CA, has now transitioned to PoS with its Ethereum 2.0 update. While Ethereum's versatility is well-documented, its performance under different conditions and its resilience to attacks like double-spending or node failure require comprehensive examination [18].

D. XRP LEDGER

The XRPL, serves as the backbone for the digital payment protocol Ripple. XRPL differentiates itself by focusing on high-speed, low-cost international transactions, using a CA called the RPCA [19]. While the XRPL's performance has been lauded, systematic empirical evaluations and comparisons with other blockchain platforms remain limited.

E. THE CONCEPT OF DOUBLE SPEND ATTACK

A double spend attack is a potential flaw in a digital cash scheme where a single digital token can be spent more than once [20]. This is possible because a digital token consists of a digital file that can be cloned or reproduced. Unlike physical tokens, such as coins or banknotes, digital tokens can be duplicated and spent in more than one place, effectively counterfeiting the digital currency. In the context of blockchain protocols, this problem is particularly challenging. This is because transactions on these networks are not always immediately committed to the ledger, creating a window of opportunity for malicious actors. During this window, an attacker can send a transaction, and before it is committed to the ledger, they send another transaction spending the same tokens but directed to a different address, typically one they control.

In an effective double spend attack, both transactions are validated, leading to a situation where the same digital tokens are spent twice, undermining the integrity of the ledger and leading to a loss of trust in the system. The XRPL, like many other blockchain protocols, is designed to mitigate the risk of double spend attacks. It achieves this through the use of the RPCA. The RPCA is designed to reach consensus among nodes on the transactions to be included in the next ledger. As a result, in an effectively functioning XRPL, a double spend attack would be identified and rejected during the consensus process. In the following sections, we describe how this attack was simulated on the XRPL client using the BBF, and the impact it had on the system's performance and validity.

F. THE CONCEPT OF NODE FAILURE OR CRASH

In distributed systems, such as blockchain protocols, node failure or crash is a frequently encountered phenomenon that can significantly impact the performance and security of the network. A node, in this context, refers to a machine or server

that participates in the network by validating and relaying transactions. Nodes play a critical role in maintaining the integrity, security, and overall functioning of the blockchain protocol. Therefore, a failure or crash involving one or more nodes could have substantial implications.

A node failure or crash, as depicted in **Fig. 2** can be defined as a sudden and unexpected termination of a node's functions and responsibilities in the network due to reasons such as hardware failure, software bugs, network disruptions, power outages, or malicious attacks. The latter condition may result in the node losing its ability to respond and propagate blocks and validate transactions, ultimately disconnecting it from the network. Two primary types of node failure exist in distributed systems: crash failures and Byzantine failures. A crash failure, occurs when a node stops working altogether, ceasing to respond to requests or perform tasks. On the other hand, a Byzantine failure is more complex and problematic. It refers to a condition where a node starts to behave erratically or maliciously, potentially sending out incorrect or conflicting information to other nodes in the network.

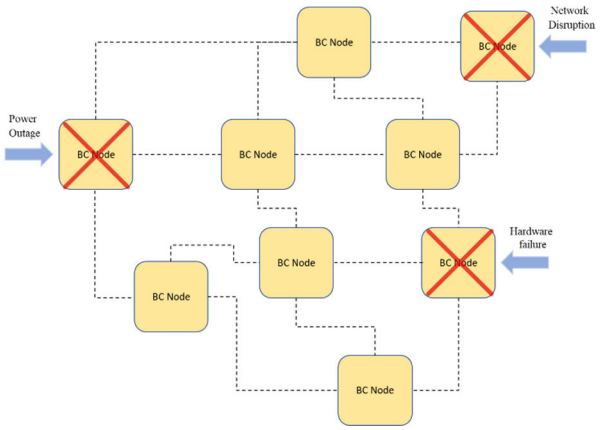

FIGURE 2. Blockchain node failure or crash.

In a blockchain protocol, node failures can influence the network's resilience, capacity, and performance. Specifically, they can affect the network's transaction throughput, latency, and ability to reach consensus, given that fewer nodes will be available to validate transactions and contribute to the consensus process. Moreover, in the context of the RPCA, used by XRPL, the failure of a single node may not significantly affect the network's overall operation due to its distributed nature. However, the simultaneous failure of multiple nodes, especially those possessing high influence in the network (like validators), can disrupt the consensus process, slow down transaction validations, and pose potential risks to network security.

G. PERFORMANCE EVALUATION METRICS

Performance evaluation of blockchain platforms can involve various metrics, including transaction throughput, latency, scalability, and network resilience, among others [21]. Transaction throughput refers to the number of transactions processed per unit of time, while latency measures the time taken to confirm a transaction. Scalability reflects the system's ability to maintain performance as the network size or load increases. Network resilience reflects the system's ability to recover and continue operating under adverse conditions, such as node failure or double-spend attacks [22].

H. REVIEW OF EXISTING RESEARCH

Existing research has examined the performance of blockchain platforms individually, often focusing on Bitcoin due to its market dominance [23]. A few studies have compared the performance of Ethereum and Bitcoin [24], while others have analyzed private blockchain platforms [25]. However, the literature lacks a comprehensive framework for systematically comparing various blockchain platforms under different conditions, a gap that the BBF aims to fill.

Existing performance evaluation tools have typically focused on one or a few specific aspects of performance, such as transaction throughput or latency [26]. While these metrics are important, they do not provide a comprehensive view of the system's performance. Moreover, most of these evaluations have been conducted under ideal network conditions, neglecting to consider the impact of adverse conditions like node failures or double-spend attacks.

III. METHODOLOGY

This section outlines our research methodology, which is focused on the application of the BBF to two specific use cases: Ripple XRPL and Ethereum. The BBF, as detailed in Section II, provides a robust platform for evaluating blockchain protocols towards the validation of the design decisions of different blockchain protocols. In the following sections, we describe the practical application of the BBF, demonstrating its utility in analyzing and comparing the performance characteristics of these two blockchains.

A. USE CASES METHODOLOGY

Use Case #1 - Ripple XRP Ledger:

In the initial use case, we employed the BBF to the official XRPL Docker client within a private environment. The process involved conducting an empirical assessment designed to stress test the XRPL client's performance under Byzantine faults. The secure data handling and reliability of the system during this process were ensured by the BBF. Moreover, the BBF's user interface provided a consistent, user-friendly platform for executing the tests and visualizing the outcomes.

Use Case #2 - Ethereum: Applying the BBF to the official Ethereum Docker client constituted the second use case, which was similar to the XRPL use case. The BBF enabled automated deployment and secure data handling, whilst providing the flexibility to customize stress testing for the evaluation of Ethereum network's performance under various conditions. The BBF's user interface-maintained consistency

in terms of visual and functional elements for executing tests and visualizing outcomes.

In selecting XRPL and Ethereum as our primary use cases, we aimed to test the BBF across diverse blockchain technologies. XRPL, with its unique CA and focus on payment protocols, and Ethereum, known for its general-purpose functionality and various consensus methods, present contrasting yet significant instances within the blockchain domain. It's crucial to note that our use of XRPL and Ethereum involved their official Docker clients, ensuring that our tests were conducted on authentic and representative software from these platforms. This strategic choice allowed for a comprehensive examination of the BBF's capabilities across different types of blockchain implementations. Furthermore, the support from the University Blockchain Research Initiative (UBRI) and the Ripple community made XRPL a particularly suitable candidate for this study. The decision to focus on these two platforms, rather than a larger set, was driven by the objective of conducting a detailed, comparative analysis, enriching the research findings and ensuring a robust testing environment for the BBF.

B. EXPERIMENTAL EVALUATION

For both use cases, experimental assessments were carried out using the BBF within an Amazon AWS EC2 instance (c5.2xlarge), comprised of 8 vCPUs, 16 GiB of Main Memory/DRAM, and up to 10 Gbps Network Bandwidth. The BBF was instantiated to conduct stress tests to evaluate the performance of the blockchain clients under Byzantine faults.

It is important to clarify that our experiments did not rely on simulations, but rather involved the actual deployment of the blockchain clients for XRPL and Ethereum. This approach ensures that our findings reflect real-world test cases and not merely theoretical or simulated outcomes. While the use of a single machine configuration and a private network provided a controlled environment, it was chosen to isolate external variables and focus on the BBF's capabilities in practical scenarios.

In terms of repeatability and reproducibility, significant measures have been taken to enable other researchers to replicate our study. Our source code is open-source, fostering transparency and allowing others to verify, use, and modify it. Alongside this, we have created extensive end-to-end tutorials and thorough documentation, guiding users through every step of setting up and executing the BBF experiments. This documentation includes detailed information on software versions, hardware specifications, network settings, and methodologies for data collection, ensuring that others can accurately follow and replicate our experiments. These resources are aimed at enhancing the scientific rigor of our work and encouraging further exploration in the area of blockchain.

This methodology fostered an in-depth evaluation of the BBF and its efficacy in benchmarking two markedly different blockchain clients - XRPL and Ethereum. The findings from these use cases not only shed light on the performance characteristics of these blockchain protocols but also encourage the utility of the BBF as a powerful tool for benchmarking blockchains.

To provide a clearer understanding of the specific metrics employed by the BBF in evaluating the performance of the blockchain clients under Byzantine faults, we present Table 1. This table outlines the key metrics compared across the Ethereum PoA and XRPL RPCA use cases. These metrics, including latency, throughput, consensus time, response to double spend attack, and node failure/crash resilience, were selected for their ability to assess various aspects of blockchain performance. The table details how each metric was validated in the respective use cases, providing a comparative overview of the performance characteristics evaluated by the BBF.

IV. USE CASES

This section presents the two use-case studies that form the empirical core of this study, detailing the application of the BBF to the Ripple XRPL and Ethereum. Each subsection includes a concise introduction to the respective blockchain protocol, elaboration of the BBF application process, a summary of the data collected, and the outcomes of the subsequent analysis.

A. USE CASE #1: XRPL AND DOUBLE SPEND ATTACK

The XRPL is a blockchain-based digital payment protocol. The distinguishing feature of XRPL is its CA, which enables faster transactions compared to traditional proof-based algorithms.

The BBF was instantiated to deploy the XRPL client in a private and controlled environment. Following the BBF's processes, an array of metrics was defined, and the necessary testing methodologies were prepared. The client was then subject to a series of stress tests under Byzantine faults. The BBF's automation capabilities and secure data handling mechanisms were pivotal throughout this process, ensuring the consistent collection of reliable data. The data collected through this process was substantial, incorporating various performance metrics, such as transaction speed, scalability, and fault tolerance. This data was subsequently analyzed and interpreted to gain insights into the XRPL client's performance under stress conditions.

1) SIMULATING AND ANALYZING DOUBLE SPEND ATTACKS ON XRPL USING BBF

To simulate a double-spend attack on the XRPL client, we have first deployed a network of ten validators configured as a full mesh network, which means every validator was connected to every other. All validators were set to participate in the consensus process and were included in the so-called unique node list (UNL) [27]. This configuration was chosen to maximize the number of nodes participating in the consensus process, making the network more robust and representative.

TABLE 1. BBF key metrics comparison across use cases.

Metric/Aspect	Description	Validated in Ethereum (PoA) Use Case	Validated in XRPL (RPCA) Use Case
Latency	Time for transaction confirmation.	Yes, with noted increases during node failures.	Yes, fluctuations observed during node failures.
Throughput	Transactions processed per time unit.	Implied stable, based on operational continuity.	Implied stable, based on operational continuity.
Consensus Time	Time to reach consensus.	Implied effective, based on attack resilience.	Implied effective, based on attack prevention.
Response to Double Spend Attack	Ability to prevent double spend attacks.	Yes, effective measures in simulations.	Yes, significant resistance demonstrated.
Node Failure/Crash Resilience	Continuity in case of node failures/crashes.	Yes, maintained operational continuity.	Yes, showed resilience and fault tolerance.

The double-spend attack was simulated using a custom script implemented in Node.js[3] with the ripple-lib,[4] a Ripple client library. The script was designed to send transactions to the XRPL network. In constructing transactions for the XRPL, each transaction must include a sequence number which is derived from the sequence number of the last closed ledger, incremented by one. To simulate a double-spend attack, two transactions were crafted using the same sequence number and sent into the network. As the XRPL client was designed to reject transactions with duplicate sequence numbers, the second transaction was expectedly rejected by the consensus process. The transactions were submitted through a single node, emulating a scenario where an attacker might attempt to spend the same digital tokens twice from the same point of access.

We have then utilized the monitoring system built into the BBF to track the network's behavior during the simulated attack. The system logged the failed transactions, and by observing their hash values, it was confirmed that the transaction rejected by the network was indeed the second transaction submitted by the script. The match between the hash of the failed transaction and the hash returned by the script demonstrated the efficacy of the XRPL's defense mechanism against double-spend attacks.

2) FINDINGS: DOUBLE-SPEND ATTACK - XRPL CLIENT

Before executing the experiment, it was essential to verify that the XRPL network was operational, synchronized, and ready to process incoming transactions. To accomplish the latter, we had developed a custom script, which returned comprehensive details about the current state of the network. That information included the build version, number of complete ledgers, synchronization duration, load factor, server state, uptime, and details of the validated ledger among other key metrics. This initial step played a crucial role in setting up a successful experiment by providing a real-time overview of the network's readiness and status. The complete response from the network is depicted in **Fig. 4**.

[3]https://nodejs.org/en

[4]https://github.com/XRPLF/xrpl.js


```
{
  "TransactionType":"Payment",
  "Account":"rHb9CJAWyB4rj91VRWn96DkukG4bwdtyTh",
  "Fee":"10",
  "Destination":"rhhPSx419uscUtcGEKxcnLyrbMxCdJdoJs",
  "DestinationTag":9318,
  "Amount":"1000000000",
  "LastLedgerSequence":267,
  "Sequence":3
}
```


FIGURE 3. Signed XRPL transaction - before submission.


```
{
  buildVersion: '1.9.1',
  completeLedgers: '2979-4837',
  initialSyncDurationUs: '25407489',
  ioLatencyMs: 1,
  jqTransOverflow: '0',
  lastClose: { convergeTimeS: 2, proposers: 4 },
  load: {
    jobTypes: [ [Object], [Object], [Object], [Object], [Object],
[Object] ],
    threads: 6
  },
  loadFactor: 1,
  nodeSize: 'small',
  peerDisconnects: '4',
  peerDisconnectsResources: '0',
  peers: 8,
  pubkeyNode: 'n9KnmQPrUGFhCwMpVivugTgurmKTZrMJ7GSZ6KDJtQq9S144Q45G',
  pubkeyValidator:
'nHBVSL46zf5NKPitkQwnqugSjCPEukyrbbALMjYeklfSQkoFfRxV',
  serverState: 'proposing',
  serverStateDurationUs: '5587032154',
  stateAccounting: {
    connected: { durationUs: '24002763', transitions: '1' },
    disconnected: { durationUs: '1404725', transitions: '1' },
    full: { durationUs: '5587032154', transitions: '1' },
    syncing: { durationUs: '0', transitions: '0' },
    tracking: { durationUs: '0', transitions: '1' }
  },
  time: '2023-May-26 13:24:02.203482 UTC',
  uptime: 5612,
  validatedLedger: {
    age: 3,
    hash:
'A3DF63C0286C1F7B60A4CE57B18A04D8B11B8A7E327832DE8A34F60BAB6406A2',
    baseFeeXRP: '0.00001',
    reserveBaseXRP: '20',
    reserveIncrementXRP: '5',
    ledgerVersion: 4837
  },
  validationQuorum: 4,
  validatorList: {
    count: 1,
    expiration: '2024-May-25 11:50:00.000000000 UTC',
    status: 'active'
  },
  hostID: 'xrpl-validator-genesis'
}
```


FIGURE 4. Signed XRPL transaction - before submission.

Through the course of the experiment, the resilience of the XRPL client against double-spend attacks was thoroughly tested. The script was used to send two transactions with

the same sequence number to the network through a single node. The first transaction was accepted, while the second was immediately identified as a duplicate and rejected. To illustrate the transaction process in practice, we first examine a concrete example of a signed transaction before submitted to the network. The signed transaction – as JSON object – before submission is depicted in **Fig. 3**.

After successful submission, the transaction response is depicted in **Fig. 5**. The latter, demonstrates the lifecycle of a transaction in the XRPL network. After the transaction is prepared and signed, it's submitted to the network. The response, contains details about the submission result, including whether the transaction was successful (*resultCode: tesSUCCESS*), the validation time, and the transaction hash, which is a unique identifier of the transaction on the network etc.


```
{
  "resultCode": "tesSUCCESS",
  "resultMessage": "The transaction was applied. Only final in a
validated ledger.",
  "engine_result": "tesSUCCESS",
  "engine_result_code": 0,
  "engine_result_message": "The transaction was applied. Only final
in a validated ledger.",
  "tx_json": {
    "Account": "rHb9CJAWyB4rj91VRWn96DkukG4bwdtyTh",
    "Amount": "1000000000",
    "Destination": "rhhPSx419uscUtcGEKxcnLyrbMxCdJdoJs",
    "DestinationTag": 9318,
    "Fee": "10",
    "LastLedgerSequence": 267,
    "Sequence": 3,
    "SigningPubKey":
"0330E7FC9D56BB25D6893BA3F317AE5BCF33B3291BD63DB32654A313222F7FD020",
    "TransactionType": "Payment",
    "TxnSignature":
"304402203C9A0F33079D822D67016C592A7CC24AD32850CBF39DDC026ADBAB316789
7841022012211D742D4DD3A2233510685ED1495BAA403D51DA873F4B8145B495A3D595
325",
    "hash":
"288F25FA041D5C4842B98EEBD5AEB1D26348BCB7E9146A184340EE46C9A7FA7D"
  },
  "validation_time": "107.85130000114441"
}
```


FIGURE 5. Successful transaction - network response.

From a data analysis perspective, this experiment provided us with key insights into the working of the XRPL client's defense mechanisms against double-spend attacks. The findings confirmed that the consensus process and sequence number mechanisms were effective in detecting and rejecting the second fraudulent transaction. Furthermore, the transaction hashes and sequence numbers also confirmed the XRPL client's efficacy in identifying and preventing duplicate transactions. Performance measurements during the attack scenario showed no significant impact on the XRPL network's latency, throughput, or consensus time, indicating a high degree of resilience against double-spend attacks. This is crucial as the ability to maintain steady performance, even when under attack, is a key characteristic of a robust blockchain protocol.

Moreover, the findings also demonstrated the usefulness of the BBF monitoring system in identifying and documenting the network's response to attempted double-spend attacks. The system's capacity to gather real-time data, including transaction hashes and sequence numbers, played a crucial part in the ability to assess the effectiveness of the XRPL client's response mechanisms.

B. USE CASE #1: XRPL AND NODE FAILURE OR CRASH

1) SIMULATING AND ANALYZING THE NODE FAILURE OR CRASH

Understanding the behavior of a distributed ledger system like the XRPL during disruptions, such as node failures, is key to evaluating its strength and performance. Therefore, this part of the study explains how node failures were simulated and examined within an XRPL client using a set of specially developed scripts.

At first, a script was developed to manage 10 XRPL validators, all running in Docker containers. This script mimics node crashes and recoveries by randomly stopping a validator and then restarting a previously stopped one. A particular validator, "*xrpl-validator-genesis*", is kept running throughout to maintain network continuity. The status of each validator, whether running or stopped, is logged into a Comma Separate Values (CSV) file for future analysis. At the same time, another script sends a predetermined number of transactions to the network during these disruptions. This gives a more thorough understanding of how the network might behave under these conditions.

An additional script was created to record the processing time for each transaction. This script notes the time taken for each transaction and logs this data, with a timestamp, into another CSV file. The data includes the transaction number, the time it was sent, and the time it took to process. In short, the two scripts together provide a thorough method for simulating node failures in an XRPL network and analyzing the results. Using this method can offer valuable insights into the network's tolerance to faults, possible weak points, and overall capacity to handle disruptions, thus setting up a robust framework for more research and testing.

2) FINDINGS: NODE FAILURE OR CRASH – XRPL CLIENT

In the context of assessing the performance and resilience of the XRPL in the face of simulated node failures or crashes, the combination of well-defined simulation and careful data analysis rendered several valuable insights. The empirical data, as observed and recorded, provides tangible evidence on the network's behavior under node failure conditions and provides a framework for interpreting the robustness of the XRPL.

The collected data, encompassing both the state of validators (*stopped or operating*) and transaction processing durations, were appended in two CSV files. The time-series data gathered from these CSV files form the core of the empirical findings and allowed us to perform a comprehensive analysis of the XRPL's performance under the simulated scenario of node crashes.

The data from the first CSV file represent the number of running and stopped validators over time, excluding the "*xrpl-validator-genesis*" validator which was always kept running to ensure network continuity. It demonstrated the network's capacity to tolerate random crashes and recoveries without complete failure. The number of validators operational at any given point in time did show variations

owing to the node crashes and recoveries, yet a total collapse was avoided, indicating a reasonable level of fault tolerance within the system.

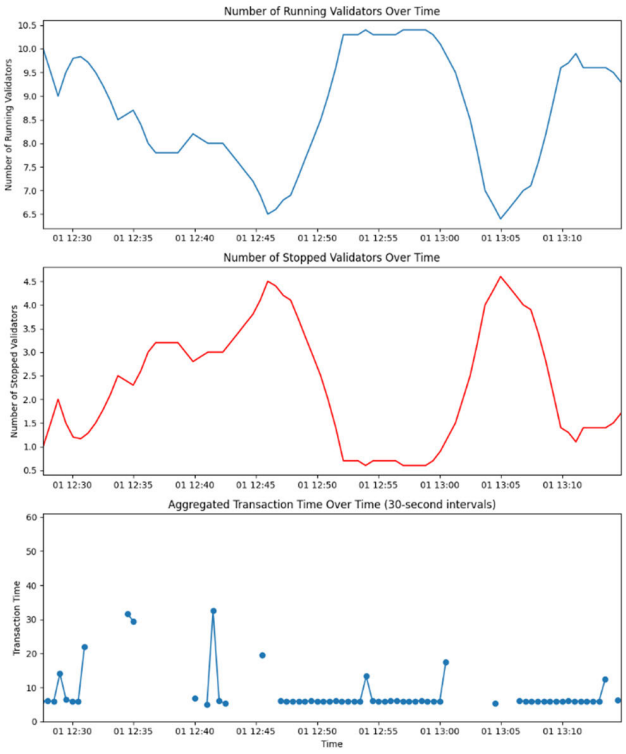

FIGURE 6. XRPL - simulating and analyzing the node failure or crash.

Further, a Python-based data visualization script provided visual plots of the behavior of validators and transaction submission times during the simulated scenario. Trends, patterns, and potential performance bottlenecks were more readily identifiable through these visual representations. For instance, while there was an observable fluctuation in the number of running validators, a minimum level of operation was maintained throughout the simulation period. Also, while there were increases in transaction times during peak disruption periods, these times remained within acceptable limits, suggesting that the XRPL can handle node disruptions without dramatically compromising on transaction processing times. Part of the Python code written to visualize the data gathered during the simulation of the node failure or crash is depicted in **Fig. 7**.

C. USE CASE #2: ETHEREUM AND DOUBLE SPEND ATTACK

Ethereum, a blockchain platform known for its smart contract functionality, uses a proof-based CA, initially PoW, and transitioning to PoS. This general-purpose platform represents a significant contrast to the XRPL and its specialized payment protocol.

Applying the BBF to the Ethereum client mirrored the process described in the XRPL use case. However, due to the differences in CAs and overall functionality, the stress tests for Ethereum were tailored to reflect these unique characteristics. Again, the BBF's automated deployment and secure data handling ensured the validity and reliability of the data collected.

The data encompassed numerous performance attributes including smart contract execution speed, scalability under various network conditions, and the network's resilience to faults. The data were then analyzed to evaluate Ethereum's performance under the conditions simulated by the BBF. The analysis showed that Ethereum's transition from PoW to PoS brought about marked improvements in scalability, despite some trade-offs in terms of decentralization. This use case demonstrated also the BBF's utility in assessing and comparing different blockchain protocols, contributing to a comprehensive understanding of Ethereum's characteristics.

Algorithm 1 Visualizing Validator and Transaction Data

```
Import pandas as pd, matplotlib.pyplot as plt
Load data:
validator_data ← Read CSV from path
transaction_data ← Read CSV from path
Convert Timestamp columns to datetime:
validator_data['Timestamp'] ← pd.to_datetime(validator_data['Timestamp'])
transaction_data['Timestamp'] ← pd.to_datetime(transaction_data['Timestamp'])
Set rolling window size to 10
Compute rolling averages:
running_avg ← rolling average of validator_data['RunningValidators']
stopped_avg ← rolling average of validator_data['StoppedValidators']
Extract float from transaction_data['Real']
Aggregate transaction data into 30-second intervals
Determine x-axis range from validator data Timestamp
Determine y-axis range for transaction time
Create 3 subplots
Plot running validators, stopped validators, and aggregated transaction time
Adjust subplot spacings
Display the combined plots
```

FIGURE 7. Visualizing XRPL node failure or crash - python code.

1) SIMULATING AND ANALYZING DOUBLE-SPEND ATTACKS ON ETH USING BBF.

This section outlines the step-by-step process employed in an experiment aimed at simulating a double-spend attack on the Ethereum network using the BBF. The experiment was executed within a controlled environment using a private Hyperledger Besu [28] client.

a: SYSTEM SETUP AND TEST CONNECTIVITY

The blockchain was set up using Docker, composed of four Ethereum nodes functioning as validators within the same network. Each node was allocated a unique IP address and port for internal communication. Network connectivity was verified via a custom script designed to check the connection status among all nodes, ensuring the effectiveness of communication channels within the network. The script's algorithm used for testing the interconnectivity of the network's validators is depicted in **Fig. 8**.

b: NETWORK PARTITIONING

To simulate a double-spend scenario, the network was partitioned into two halves, thereby creating a split-brain situation. Segment 1 contained the first two validators, while Segment

2 housed the remaining ones. This partitioning was achieved using a custom networking script responsible to manipulate the interconnection of the validators.

c: DOUBLE-SPEND TRANSACTION SIMULATION

Post partitioning, identical transactions were simultaneously submitted to a validator in each segment. Due to the disconnect between the two network segments, both transactions were considered valid and added to their respective blockchains, essentially creating a double-spend scenario.

d: RECONNECTION AND CONSENSUS RESOLUTION

Following the transaction submission, the network was restored by reconnecting the split nodes, thus recreating a conflicting blockchain scenario. Ethereum's PoA CA was leveraged to resolve this conflict. On re-establishment of network connectivity, the conflicting transactions were identified, and one version was discarded based on the consensus protocol, thereby preventing the double-spend situation and maintaining the network's integrity.

Algorithm 2 Checking Validator Connections

```
Initialize array of validators
for each validator i in validators do
    for each validator j in validators do
        if i ≠ j then
            if validator i can connect to validator j then
                Print "Connection from validator i to validator j is open."
            else
                Print "Connection from validator i to validator j is closed."
            end if
        end if
    end for
end for
```

FIGURE 8. Ethereum network - test connectivity between validators.

2) FINDINGS: DOUBLE-SPEND ATTACK - ETHEREUM

This part of the research set out to empirically test the resilience of Ethereum against double-spend attacks. The investigation leveraged a private Ethereum network, powered by Hyperledger Besu and hosted in a Docker environment. Four validator nodes were configured, each running in its Docker container. The experimental setup followed the PoA consensus model where the network can tolerate at most (N-1)/3 faulty nodes. We exploited the partition tolerance property of the network, splitting it into two segments: one containing two validators and the other containing the remaining two.

To perform the double-spending attack, two identical transactions were crafted, each aimed at spending the same Ether funds from a particular address. In the partitioned state, these transactions were sent simultaneously to validators in separate network segments. Since the validators had no means of communicating due to the network split, they could not reach a consensus on the transactions' legitimacy.

Observations during the experiment included the successful submission of duplicate transactions and the subsequent states of these transactions upon network reconnection. Notably, despite the network split, only one of the transactions was executed successfully when network connectivity was reestablished. This result highlights Ethereum's resilience to double-spending attacks, an attribute that can be largely attributed to its implementation of the PoA CA. In the absence of a consensus (as was the case during the network partition), the protocol defaults to a state of safety, rejecting conflicting transactions until consensus can be restored.

Our findings align with theoretical expectations and further underscore Ethereum's robustness against double-spending attacks. This resilience significantly enhances Ethereum's network security, and the credibility of transactions executed on its blockchain. However, it is worth noting some limitations of the experiment. The testing environment was a simplified representation of Ethereum's real-world ecosystem. The setup with just four validator nodes may not fully emulate the complexity of a public Ethereum network where thousands of nodes participate.

Additionally, the use of the BBF in executing the simulation scenario added another layer of robustness to our findings. The framework allowed for the structured and repeatable execution of the double-spending scenario, minimizing the potential for human error, and ensuring the consistency of the experimental conditions. Using the BBF, we could accurately and systematically adjust the network parameters, submit transactions, and track their processing status. This level of precision facilitated a thorough analysis of Ethereum's response to the double-spending attack, underscoring the reliability of the findings.

In this regard, the benchmarking framework proved instrumental in validating our findings, thereby contributing to their potential applicability in real-world contexts. Its use highlights the importance of systematic tools in conducting blockchain research and the valuable insights that such methodical approaches can yield.

D. USE CASE #2: ETHEREUM AND NODE FAILURE OR CRASH

1) SIMULATING AND ANALYZING THE NODE FAILURE OR CRASH SCENARIO

To evaluate the resilience of the Ethereum network and examine its behavior under unexpected conditions, we have conducted simulations to model scenarios where validator nodes experienced unpredictable crashes or stops. This process was vital in determining the network's robustness in the face of node failures and evaluating the impact these crashes had on transaction performance.

The simulation process comprised of a Python script – **Fig. 9 -** that simultaneously initiated transactions to the Ethereum network and randomly terminated nodes to simulate failures. The script was specifically designed to capture the dynamic state of the Ethereum network, sending a series of transactions and observing the behavior during node crashes. Transaction details included aspects such as transaction number, timestamp of the transaction initiation, time taken for the transaction to complete, and the transaction

status. These details were monitored and appended into a CSV file with the following structure:

- **Transaction**: The respective transaction number.
- **Timestamp**: The time at which the transaction transpired.
- **Time**: The time taken to complete the transaction.
- **Status**: The transaction's status, signifying whether it was successful or had failed.

At the same time, the state of the nodes (*running or stopped*), as depicted in **Fig. 10**, was captured and appended into a dataset with the following structure:

- **Timestamp**: This represented the specific time when the observation was recorded.
- **Running Validators**: This indicated the count of validators that were active at each logged timestamp.
- **Stopped Validators**: This signified the count of validators that had halted at each recorded timestamp.

Algorithm 3 Execute TXs in Ethereum Network

```
procedure InitializeWeb3Connection
    Set PRIVATE_KEY, ACCOUNT_ADDRESS from PRIVATE_KEY, TO_ADDRESS
end procedure
procedure SendTransactions(NUM_TXNS)
    Open "transactions_time.csv" for writing with headers
    for i from 1 to NUM_TXNS do
        Setup transaction details
        Sign the transaction
        try
            Send transaction, wait for receipt
            Write to CSV: transaction details, "Successful"
            catch  Timeout or Exhausted
                Write to CSV: transaction details, "Failed"
            end catch
            Wait 1 second


            procedure Main
                NUM_TXNS ← User input
                CALL SendTransactions(NUM_TXNS)
            end procedure
```

FIGURE 9. Python script - Execute TXs in ethereum network.

Two plots were constructed to visualize the results:

Number of Running and Stopped Validators Over Time -Fig.11-: This plot visually demonstrates the fluctuations in the number of running and stopped validators over time. Time was represented on the x-axis, while the number of validators was displayed on the y-axis. A rolling average with a window size of 3 was utilized to smooth the data points and render a clearer trend over time.

Transaction Times Over Time -Fig.12-: This plot depicted transaction times in correlation to their occurrence time. Each data point represented a transaction and was illustrated as a scatter plot. The use of color-coding facilitated the identification of transaction statuses: successful transactions were represented in green, failed ones in red, and timed-out transactions in orange.

2) FINDINGS: NODE FAILURE OR CRASH – ETHEREUM CLIENT

In our study, we simulated a situation where parts of the Ethereum blockchain protocol were failing, to understand how resilient the network is during such disruptions. Specifically, we observed how many nodes were active or inactive

Algorithm 4 Validator Management Procedure

```
procedure InitializeCsv
    Write "Timestamp,RunningValidators,StoppedValidators" to validators.csv
end procedure
procedure ManageValidators
    Initialize VALIDATORS array with validator names
    Initialize STOPPED array as empty
    while True do
        if Number of VALIDATORS > 3 then
            Pick a random validator from VALIDATORS
            Stop the selected validator using Docker
            Add the stopped validator to STOPPED array
            Remove the stopped validator from VALIDATORS array
        end if
        if Random number is odd and STOPPED array is not empty then
            Pick a random number of validators from STOPPED to restart
            for all validators in the selected number to restart do
                Pick a random validator from STOPPED
                Start the selected validator using Docker
                if Validator started successfully then
                    Add validator to VALIDATORS array
                    Remove validator from STOPPED array
                end if
            end for
        end if
        Timestamp ← Current time
        Write Timestamp, size of VALIDATORS, size of STOPPED to validators.csv
        Sleep for 30 seconds
    end while
end procedure
procedure Main
    CALL InitializeCsv()
    CALL ManageValidators()
end procedure
```

FIGURE 10. Simulate node crash scenario on ethereum network.

during these failures. Our findings showed that the number of active and inactive nodes varied due to these disruptions. We also kept detailed records of each transaction during this period. This included information like the transaction's unique number, the exact time it occurred, how long it took to process, and whether it was successfully completed or not. This data is crucial for understanding how well the Ethereum network can handle unexpected problems and maintain its performance.

In the course of multiple simulations, a consistent trend emerged regarding Ethereum's network behavior under node failure scenarios. After each node failure, the network exhibited a predictable recovery pattern, with transaction processing times eventually stabilizing after a period of increased failures and delays. This recurring pattern led us to define the endpoint for each simulation once the network's performance reached a constant state post-recovery. This point is characterized by a return to normal levels of transaction processing times and success rates. This decision allowed us to focus on the immediate impacts of node failure and the subsequent recovery, thereby providing a comprehensive perspective on Ethereum's resilience and adaptability under such adverse conditions.

While observing the trend in the corresponding dataset, it was clear that the Ethereum network showcased impressive resilience in the face of node failures. Despite the sudden disruptions of validators, the network was able to maintain its functionality, underlining the redundancy and robustness of decentralized blockchain protocols. The analysis of transaction data revealed important insights into transaction performance under these conditions. As node failures

increased, there was a notable effect on transaction times and success rates. It was observed that the rate of transaction failure and timeout events slightly increased during periods of high node failure, hinting at the network's strain under these conditions. However, the network still managed to process a majority of the transactions successfully, which testifies to Ethereum's robustness.

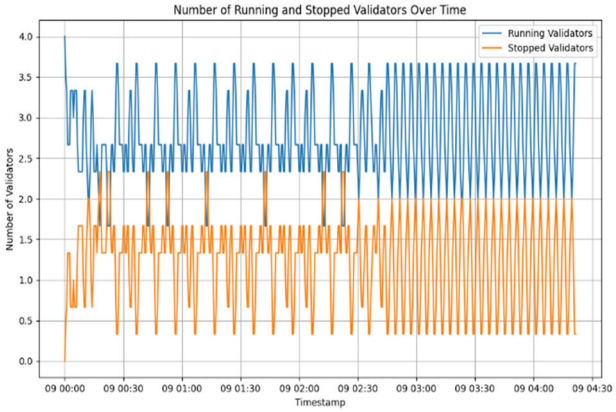

FIGURE 11. Ethereum node crash scenario - validator status over time.

The graphical representation of the data gave a clear depiction of the network's behavior. The plots in **Fig. 11** and **Fig. 12** are showing the number of running and stopped validators over time effectively illustrated the network's dynamic state during the simulation. The scatter plot of transaction times demonstrated the impact of node failure on transaction performance. It displayed an increase in transaction times as well as transaction failure events during periods of high node failure.

The empirical data collected, and the subsequent analysis underlined the Ethereum network's resilience and robustness in the face of node failure. Even though the node failures led to a slight increase in transaction times and failure rates, the network maintained its operational integrity and processed most transactions successfully. These findings are critical in understanding the behavior and reliability of the Ethereum network under unpredictable and adverse scenarios.

V. DISCUSSION

A. ANALYSIS AND DISCUSSION OF USE CASE #1

The experimental investigation undertaken in this study focused on understanding the performance and validity of the XRPL client under the impact of Byzantine faults—specifically, double-spend attacks and node failure or crash scenarios. To facilitate this investigation, the BBF was utilized as a key tool. These experiments revealed empirical data and insights into the XRPL's robustness, resilience, and performance characteristics under adversarial conditions. The use of the BBF was critical in simulating these scenarios and in gathering detailed performance metrics.

In the context of the double-spend attack, the XRPL client demonstrated significant resistance. Transactions that attempted to double-spend were consistently detected and prevented from being included in the validated ledger. This robustness against double-spend attacks can be attributed to the consensus protocol employed by XRPL, which emphasizes strict transaction ordering and validation. This validation, coupled with the uniqueness of the transaction sequence numbers for each account, ensures that double-spend transactions are effectively detected and rejected, bolstering the network's integrity and security.

In terms of node failure or crash scenarios, the XRPL client exhibited resilience and fault tolerance. Despite the randomized crashing and recovery of nodes, the network maintained operational continuity and processed transactions without encountering catastrophic delays or failures. This resilience against node crashes can be interpreted as an affirmation of the distributed, decentralized nature of the XRPL network, where the system can withstand individual node failures without compromising overall network performance.

Additionally, our analysis examines how XRPL's unique CA, influences its transaction speed and energy efficiency. We observed that RPCA's streamlined approach significantly contributes to XRPL's high transaction speed, maintaining rapid consensus times even during stress conditions like double-spend attacks and node failures. This efficiency not only accelerates transaction processing but also implies reduced energy consumption compared to more computation-intensive CAs.

However, it is essential to note that the network performance was observed to experience fluctuations under node failure scenarios. Increases in transaction times during peak disruption periods suggested that such disruptions could impact network throughput. Nonetheless, these variations remained within acceptable limits, demonstrating that the XRPL can manage node disruptions without significantly compromising transaction processing times.

The implications of these findings for the XRPL network and its CA are multifold. First, the demonstrated robustness against double-spend attacks underscores the effectiveness of the XRPL's consensus protocol in maintaining network security. Second, the observed resilience against node failures reflects the inherent fault tolerance of the XRPL network. Together, these findings suggest that the XRPL, under its current design and CA, possesses considerable resistance to common Byzantine faults, further solidifying its potential for robust, decentralized financial transactions.

Finally, reflecting on the effectiveness of the BBF in benchmarking the XRPL client's performance, it is evident that the BBF served as a powerful tool in simulating adversarial conditions and assessing the XRPL client's response. It enabled the systematic execution of Byzantine faults, facilitated the collection of empirical data, and allowed the evaluation of the XRPL client's behavior under such conditions.

In summary, the findings of these experiments underscore the robustness, resilience, and fault tolerance of the XRPL client under Byzantine faults. While certain performance

fluctuations were observed under node failure conditions, the XRPL's ability to maintain operational continuity and transaction processing effectively illustrates its potential in the context of decentralized financial systems. The BBF has proven to be an effective benchmarking tool in this context, enabling a systematic and insightful evaluation of the XRPL client's performance.

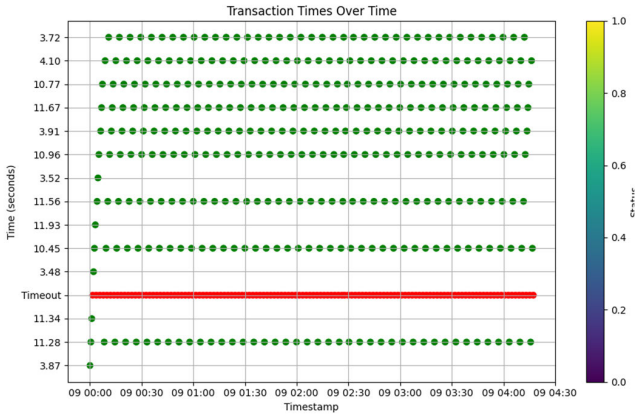

FIGURE 12. Ethereum node crash scenario - transactions' processing time over time.

B. ANALYSIS AND DISCUSSION OF USE CASE #2

The second use case provided an in-depth exploration of the Ethereum network, leveraging the Ethereum client for a range of simulations and analyses. This use case focused on gaining an understanding of Ethereum's operational characteristics, network dynamics, and performance under various conditions.

A key aspect of our analysis was the evaluation of Ethereum's CA and its impact on the network's performance. The PoS CA, as implemented in Ethereum 2.0, is designed to be less energy-intensive than the traditional PoW approach, thereby addressing some of the environmental concerns associated with blockchain technologies. Furthermore, our findings suggest that this transition has the potential to enhance Ethereum's scalability and transaction processing speed, contributing to improved overall network performance.

The first stage of this use case involved the deployment of a private Ethereum network, achieved through setting up and configuring a local Ethereum client. The BBF was employed during this stage to assess and optimize the performance of the network. Utilizing Hyperledger Besu, an open-source Ethereum client, a network of Ethereum nodes was established. The integration of BBF in this process allowed for a comprehensive evaluation of the network's capabilities. This set-up process underscored the flexibility and ease-of-use of the latter, making it a fitting choice for developing applications on the Ethereum blockchain.

The next focus of the study was on investigating Ethereum's transaction performance. A script was developed to send a specified number of transactions from one account to another, allowing the examination of transaction throughput and time taken for each transaction. Results indicated that Ethereum was capable of handling a substantial number of transactions, with the time taken for each transaction varying based on network conditions, gas prices, and transaction load. It's important to note that the study was conducted on a private network, which would have different performance characteristics compared to the public Ethereum network due to the absence of real-world transaction traffic and network congestion.

The resilience of the Ethereum network in the face of node crashes was the next area of investigation. The research simulated scenarios wherein validator nodes experienced random crashes or stops. These simulations, coupled with concurrent monitoring of the Ethereum network, provided valuable insights into the network's response to node failure and its subsequent impact on transaction performance. The analysis highlighted Ethereum's robustness and ability to function despite abrupt node stops, confirming the redundancy and robustness of decentralized blockchain protocols. However, it was observed that as node failures increased, there was a slight impact on transaction times and success rates, demonstrating that while Ethereum can handle node failures, its performance could be affected under such scenarios.

Comparing Ethereum with the previously investigated XRPL, it's clear that both have distinct operational characteristics and strengths. While the XRPL boasts of superior transaction speed and minimal transaction costs, Ethereum's strength lies in its smart contract functionality and the robustness of its network. Ethereum's handling of node failure scenarios underlines the resilience of decentralized blockchain protocols and offers valuable insights for blockchain application developers and network operators. It's also worth noting that Ethereum, being Turing-complete, provides greater flexibility for developing complex decentralized applications compared to XRPL.

This use case, through detailed simulations and performance analyses, provided comprehensive insights into Ethereum's operational dynamics and its performance under varied scenarios. It emphasized the importance of understanding the distinct characteristics and capabilities of different blockchain platforms, which can inform the choice of platform based on the specific requirements of a blockchain-based application or solution.

VI. CONCLUSION

This article, primarily aimed at testing the BBF firstly introduced in [8]. In reflecting on the findings of this study, it is imperative to reiterate that our investigation was centered on validating the functionality of the BBF, rather than conducting a performance evaluation of the XRPL and Ethereum blockchain networks. Our aim was to demonstrate the BBF's capabilities in a controlled setting, underscoring its utility for pre-launch blockchain protocol assessments.

The BBF has proven to be an effective tool for evaluating and benchmarking the performance of blockchain protocols,

demonstrated through its application to two use cases: XRPL and Ethereum. Importantly, these evaluations were conducted using the official Docker clients of XRPL and Ethereum in controlled test environments. The study provided valuable insights into the operational characteristics of these blockchains, particularly how different CAs influence network performance, transaction speed, and energy efficiency. These findings deepen our understanding of blockchain technology and offer crucial data for those looking to leverage blockchain protocols in their systems and applications.

Based on our findings, it is recommended that organizations consider the unique characteristics of different blockchain platforms in relation to their specific use cases. For example, XRPL, known for its high transaction speed and resilience against double-spend attacks, may suit applications requiring rapid processing and high security, like payment systems. Ethereum, with its smart contract capabilities and scalability, is ideal for complex transaction logic and large-scale deployment. This guidance can help stakeholders make informed decisions about platform selection, optimizing for factors such as transaction speed, security, and energy efficiency. It's important to note that these recommendations are based on the performance of the XRPL and Ethereum Docker clients as tested within the controlled environments of our study and may not fully reflect their performance in the broader, live ecosystems.

Additionally, the BBF's results should be tailored to specific organizational needs and decision-making processes. When choosing a blockchain protocol, factors like transaction nature, security level, scalability, and energy efficiency should be considered. The BBF provides insights into how different platforms perform across these dimensions, aiding in making more informed choices.

The BBF's contribution to the blockchain field addresses the gap in standard tools for comparing and evaluating different blockchain protocols. This tool supports informed decision-making, helping organizations choose the most appropriate protocol for their specific use cases based on objective performance metrics.

However, the BBF was applied to only two blockchain protocols in this study. While XRPL and Ethereum represent a wide range of characteristics, they do not cover the entire diversity within the blockchain ecosystem. Future research should broaden the BBF's application scope, testing it against a more diverse range of blockchain protocols and enhancing it to cover a broader range of performance metrics. This expansion will solidify the BBF's role as a crucial tool in blockchain technology exploration and development.

REFERENCES

[1] *Blockchain: Blueprint for a New Economy—Melanie Swan—Google Books*. Accessed: Sep. 6, 2023. [Online]. Available: https://books.google.com.cy/books?hl=en&lr=&id=RHJmBgAAQBAJ&oi=fnd&pg=PR3&dq=Blockchain:+Blueprint+for+a+new+economy&ots=XRzEA2YNg3&sig=LvraRHGvjbNzRfBpViWQQZrU8b8&redir_esc=y#v=onepage&q=Blockchain%3A%20Blueprint%20for%20a%20new%20economy&f=false

[2] I. Mistry, S. Tanwar, S. Tyagi, and N. Kumar, "Blockchain for 5G-enabled IoT for industrial automation: A systematic review, solutions, and challenges," *Mech. Syst. Signal Process.*, vol. 135, Jan. 2020, Art. no. 106382, doi: 10.1016/j.ymssp.2019.106382.

[3] K. John, M. O'Hara, and F. Saleh, "Bitcoin and beyond," *Annu. Rev. Financ. Econ.*, vol. 14, pp. 95–115, Nov. 2022, doi: 10.1146/annurev-financial-111620-011240.

[4] D. Saingre, T. Ledoux, and J.-M. Menaud, "BCTMark: A framework for benchmarking blockchain technologies," in *Proc. IEEE/ACS 17th Int. Conf. Comput. Syst. Appl. (AICCSA)*, Nov. 2020, pp. 1–8, doi: 10.1109/AICCSA50499.2020.9316536.

[5] Y. Merrad, M. H. Habaebi, E. A. A. Elsheikh, F. E. M. Suliman, M. R. Islam, T. S. Gunawan, and M. Mesri, "Blockchain: Consensus algorithm key performance indicators, trade-offs, current trends, common drawbacks, and novel solution proposals," *Mathematics*, vol. 10, no. 15, p. 2754, Aug. 2022.

[6] H. Rastogi, "An overview of blockchain technology: Architecture and consensus protocols," in *Smart City Infrastructure: The Blockchain Perspective*. Hoboken, NJ, USA: Wiley, Feb. 2022, pp. 293–315, doi: 10.1002/9781119785569.ch12.

[7] M. M. Queiroz, R. Telles, and S. H. Bonilla, "Blockchain and supply chain management integration: A systematic review of the literature," *Supply Chain Manage.*, vol. 25, no. 2, pp. 241–254, Feb. 2020, doi: 10.1108/SCM-03-2018-0143.

[8] M. Touloupou, M. Themistocleous, E. Iosif, and K. Christodoulou, "A systematic literature review toward a blockchain benchmarking framework," *IEEE Access*, vol. 10, pp. 70630–70644, 2022, doi: 10.1109/ACCESS.2022.3188123.

[9] Y. Lee, B. Son, S. Park, J. Lee, and H. Jang, "A survey on security and privacy in blockchain-based central bank digital currencies," *J. Internet Services Inf. Secur.*, vol. 11, no. 3, pp. 16–29, Aug. 2021, doi: 10.22667/JISIS.2021.08.31.016.

[10] F. Guo, G. Shen, Z. Huang, Y. Yang, M. Cai, and L. Wei, "DABAC: Smart contract-based spatio-temporal domain access control for the Internet of Things," *IEEE Access*, vol. 11, pp. 36452–36463, 2023, doi: 10.1109/ACCESS.2023.3257027.

[11] D. Mitrea, T. Cioara, and I. Anghel, "Privacy-preserving computation for peer-to-peer energy trading on a public blockchain," *Sensors*, vol. 23, no. 10, p. 4640, May 2023, doi: 10.3390/s23104640.

[12] M. Themistocleous, "Justifying the decisions for EAI implementations: A validated proposition of influential factors," *J. Enterprise Inf. Manage.*, vol. 17, no. 2, pp. 85–104, Apr. 2004, doi: 10.1108/17410390410518745.

[13] S. Nakamoto. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System*. Accessed: Mar. 5, 2021. [Online]. Available: https://www.bitcoin.org

[14] G. S. Panova and D. V. Panov, "Digital economy financial applications," in *Proc. Int. Conf. Data Sci. Appl.*, vol. 288, 2022, pp. 783–799, doi: 10.1007/978-981-16-5120-5_59.

[15] M. Touloupou, K. Christodoulou, A. Inglezakis, E. Iosif, and M. Themistocleous, "Benchmarking blockchains: The case of XRP ledger and beyond," in *Proc. 55th Hawaii Int. Conf. Syst. Sci.*, Jan. 2022, doi: 10.24251/hicss.2022.730.

[16] M. Touloupou, K. Christodoulou, A. Inglezakis, E. Iosif, and M. Themistocleous, "Towards a framework for understanding the performance of blockchains," in *Proc. 3rd Conf. Blockchain Res. Appl. Innov. Netw. Services (BRAINS)*, Sep. 2021, pp. 47–48, doi: 10.1109/BRAINS52497.2021.9569810.

[17] V. Buterin, "A next generation smart contract & decentralized application platform," White Paper, 2014, vol. 3, no. 37, p. 2-1.

[18] *Ethereum: A Secure Decentralised Generalised Transaction Ledger*, Ethereum Project Yellow Paper, 2014, vol. 151, no. 2014, pp. 1–32.

[19] *The Ripple Protocol Consensus Algorithm | Semantic Scholar*. Accessed: Mar. 11, 2022. [Online]. Available: https://www.semanticscholar.org/paper/The-Ripple-Protocol-Consensus-Algorithm-Schwartz-Youngs/bff4ecdd2c40bb67abab8d49e99c81287a7b2810

[20] D. Chaum, "Blind signatures for untraceable payments," in *Advances in Cryptology*. Boston, MA, USA: Springer, 1983, pp. 199–203, doi: 10.1007/978-1-4757-0602-4_18.

[21] N. Six, N. Herbaut, and C. Salinesi, "Blockchain software patterns for the design of decentralized applications: A systematic literature review," *Blockchain, Res. Appl.*, vol. 3, no. 2, Jun. 2022, Art. no. 100061, doi: 10.1016/j.bcra.2022.100061.

[22] E. Androulaki et al., "Hyperledger fabric: A distributed operating system for permissioned blockchains," in *Proc. 13th EuroSys Conf.*, vol. 18. New York, NY, USA: Association for Computing Machinery, Apr. 2018, pp. 1–15, doi: 10.1145/3190508.3190538.

[23] J. W. Kirkwood, "From work to proof of work: Meaning and value after blockchain," *Crit. Inquiry*, vol. 48, no. 2, pp. 360–380, Jan. 2022, doi: 10.1086/717303.
[24] F. Wilhelmi, L. Giupponi, and P. Dini, "Analysis and evaluation of synchronous and asynchronous FLchain," *Comput. Netw.*, vol. 218, Dec. 2022, Art. no. 109390, doi: 10.1016/j.comnet.2022.109390.
[25] F. Tschorsch and B. Scheuermann, "Bitcoin and beyond: A technical survey on decentralized digital currencies," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 3, pp. 2084–2123, 3rd Quart., 2016.
[26] B. Cao, Z. Wang, L. Zhang, D. Feng, M. Peng, L. Zhang, and Z. Han, "Blockchain systems, technologies, and applications: A methodology perspective," *IEEE Commun. Surveys Tuts.*, vol. 25, no. 1, pp. 353–385, 1st Quart., 2023, doi: 10.1109/COMST.2022.3204702.
[27] I. Amores-Sesar, C. Cachin, and J. Micic, "Security analysis of ripple consensus," in *Proc. Leibniz Int. Informat.*, vol. 184, Nov. 2020, doi: 10.4230/LIPIcs.OPODIS.2020.10.
[28] Hyperledger. *About Hyperledger*. Accessed: Jun. 15, 2023. [Online]. Available: https://wiki.hyperledger.org/groups/pswg/performance-and-scale-wg

MARIOS TOULOUPOU received the B.Sc. and master's degrees in digital systems from the Department of Digital Systems, University of Piraeus. He is currently pursuing the Ph.D. degree with the University of Nicosia, Cyprus. His bachelor's thesis entitled "Three-Dimensional Web Platform for Multimedia Data Management (Applying the Approach to the Data of the Holy Sepulcher Restoration Project)," where he delivered a set of innovative mechanisms that focus on "Data as a Service" technology to integrate big data management into cloud environments. During the master's study, he followed the direction of "Advanced Information Systems and Services." His master's thesis topic was "5G and V&V: Estimation of Performance in 5th Generation Network Services–Targeted Validation and Verification Tests." His Ph.D. thesis topic is "Analysis and Optimization of Consensus Algorithms in Decentralized networks." Since 2017, he has been a Researcher with the Research Center of the University of Piraeus, having high interest on 5G/SDN networks, and focusing on the technology aspects and real life events–by adopting innovative tools that meet the needs of ICT industry. Currently, he is a Researcher with the Institute for the Future (IFF), University in Nicosia. He has participated and contributed to research projects (e.g., 5GTANGO) realized in the context of EU programs. His research interests include data management throughout the software life cycle and the enforcement of service quality through the monitoring of resources in full distributed systems. His current research area is distributed ledger technologies (DLTs), while he is participating in the Ripple Research Program (An open source and peer-to-peer decentralized platform that allows for a seamless transfer of money in any form, whether USD, Yen, Litecoin, or bitcoin).

KLITOS CHRISTODOULOU received the B.Sc. degree in computer science and the M.Sc. degree in advanced computer science, with a specialization in advanced applications from The University of Manchester, U.K., and the Ph.D. degree in computer science from the School of Computer Science, The University of Manchester, in 2014. He is a Faculty Member with the Department of Management and MIS–Digital Currency, University of Nicosia (UNIC), where he has also been a Research Faculty Member with the Institute for the Future (IFF), since 2018. He has been an Adjunct Staff Member of the Information Management Group (IMG), School of Computer Science, The University of Manchester, where he has engaged in various research and teaching activities. He teaches a course on blockchain applications under UNIC's M.Sc. in a Digital Currency Program. He has provided numerous invited talks and tutorials on blockchain technologies. His research interests include data management challenges, focusing on machine learning techniques and distributed ledger technologies, with an emphasis on blockchain ledgers. He has served in the program committee at a variety of conferences. Currently, he serves as an Associate Editor for *Frontiers in Blockchain* and a Guest Editor for the Special Issue of the Future Internet (MDPI journal) on Blockchain Applications.

MARINOS THEMISTOCLEOUS received the bachelor's degree in computer science and the M.Sc. degree in information systems management from the Athens University of Economics and Business, Athens, Greece, and the Ph.D. degree in information systems integration and the Postgraduate degree in teaching and learning in higher education from Brunel University, London, U.K. He is the Associate Dean of the School of Business and one of Directors of the Institute For Future (IFF), University of Nicosia, Cyprus. He holds a certification in blockchain, FinTech, and future commerce from the Massachusetts Institute of Technology (MIT), Massachusetts, CA, USA. He teaches at the world-leading Digital Currency Postgraduate Program, University of Nicosia. He has developed blockchain applications in the areas of energy and healthcare, and serves as a blockchain advisor. He retains close relationships with industry and a consultant in areas, such as blockchain, e-business, e-health, and information systems integration. He has collaborated with the Greek Ministry of Finance; Bank of Greece; Greek Standardization Body; Greek Federation of SMEs; ORACLE, U.K.; B3-Blockchain Business Board, U.K.; Intelen, U.S.; BTO Research, Italy; Cyprus National Betting Authority; and other organizations. He has authored more than 175 refereed journal and conference papers and several teaching textbooks and has received citations and awards of excellence. His research has received funding from various bodies and organizations. He is on the editorial board of academic journals and the board of prestigious international conferences. He has run minitracks, tracks, and journal special issues on blockchain. Previously, he served as the Managing Editor for the *European Journal of Information Systems* (EJIS).