billon

Performance Report:

Comparison of Unified Enterprise DLT System versus Ethereum on:

- Performance
- Scalability
- Power consumption

Tests performed under supervision of PWC

October 2021

Executive Summary

- Under the supervision of PWC's blockchain team, the **Unified Enterprise DLT System** designed by Billon Group was compared to two different configurations of enterprise BESU/Ethereum: IBFT and PoA-Clique. Comparisons were on a like-for-like basis, eliminating transaction batching and other techniques to improve performance. In all comparisons, the better of the two Ethereum configurations is cited when drawing conclusions. The results and analysis herein are provided by Billon.

- Three metrics were chosen:

    - **Performance:**

        - Throughput (transactions per second) was chosen for financial services use cases; and

        - Capacity (number of 200kB documents published on-chain) was chosen for document management use cases.

    - **Power consumption**: mWh spent per financial or document transaction

    - **Scalability**: comparison of performance and power consumption in a 100-node network and a larger 500-node network.

- Conclusions are as follows:

    - **Performance:**

        - For financial transactions, the Unified Enterprise DLT System achieved a **2.5 times higher throughput** than Ethereum with only 100 nodes, and at 500 nodes, achieved throughput of 3,000 transactions per second without batching (a technique used to bundle transactions for higher performance)

        - For documents, the Unified Enterprise DLT performed **3.2 times better** at peak loads for 500 nodes, providing the capacity to store 41.3 million "on chain" documents, i.e., 200kb high-value documents which require complete storage.

    - **Scalability:**

        - The Unified Enterprise DLT's transaction throughput **increased by 156%** (2.56x of the initial result) when using 500 nodes vs 100 nodes, and for documents, its capacity **increased by 158%** with more nodes.

        - Ethereum architecture document daily storage capacity **increased by only 19%,** while results for financial transactions were inconclusive.

    - **Power consumption:**

- At peak load and for the most demanding use case of document publication, the Unified Enterprise DLT consumed 68% less power.
    - At peak load the power consumption of the Enterprise Enterprise DLT System was **0.029 mWh** per transaction and **0.18 mWh** per document
    - Ethereum's system consumed **0.57 mWh per document**, which is **3.1 times less energy efficient** than the power consumption of the Unified Enterprise architecture.
- For the Unified Enterprise DLT, when moving from 100 nodes to 500 nodes, energy efficiency improved as **power consumption for a financial transaction fell by 39%** (i.e., 0.029 mWh/transaction at 500 nodes, vs 0.048 mWh/transaction at 100 nodes). For Ethereum, power consumption increased when moving from 100 nodes to 500 nodes, for documents and the tests were inconclusive for transaction throughput

Table of Contents

Executive Summary **2**

Administration of Tests **5**

Test objectives and Set-Up **5**

Business Rationale and Choice of Metrics **6**

Main Conclusions of Test Results **7**

On-Chain Document Management Results: 7

Financial Atomic Transaction Results 8

Details of tests results **9**

Test methodology **14**

Test setup **15**

Energy Efficiency Analysis **17**

APPENDIX 1: Smart Contracts code used for deployment **20**

APPENDIX 2: Sample reporting and measurement data **21**

APPENDIX 3: Sample monitor of key hardware resources **23**

APPENDIX 4: PWC Sample Energy Efficiency Analysis **29**

Administration of Tests

The tests were performed over the week from October 11th, 2021 to October 17th, 2021, by the employees and subcontractors of Billon, and were organized by Andrzej Horoszczak, CTO.

The tests were performed under the supervision of a blockchain team from PricewaterhouseCoopers GmbH (PwC Germany). The supervision included review of the test scripts & SC code, full access to the infrastructure as well as full access to tests' recordings. The statement of objective validity of the data was out of the supervision scope of PWC. PWC performed the full data and raw energy efficiency analysis using PWC data tools, and an extract of the analysis is Attached to this report (Appendix 4).

The results were shared with EBSI, the European Blockchain Service Infrastructure initiative, which serves 29 European Blockchain Partnership countries.

Test objectives and Set-Up

We performed in-depth benchmarking of popular blockchain architectures: Hyperledger BESU/Ethereum with two consensus protocols: PoA-Clique and IBFT2, and the Unified Enterprise DLT (designed by Billon). The test objective was to estimate energy efficiency and identify potential performance bottlenecks. Therefore each of the three blockchains was deployed on identical hardware in two scale scenarios, simulating a 500% increase in blockchain adoption:

(1) **Early Adoption Scenario (100 nodes)**: 10 countries and 10 organizations within each country, for a total of 100 participating organizations
(2) **Larger Adoption Scenario (500 nodes):** 25 countries and 20 organizations within each country, for a total of 500 organizations

During each Adoption Scenario two business scenarios were tested: Trusted Business Data & Documents Publication as well as Atomic Financial Transactions. Early scenario was run on 10 Dell PowerEdge servers with full hardware power consumption monitoring, which was the basis for calculating the energy efficiency of each of the blockchain systems both for each financial transaction and for each business data object creation. The Larger Adoption Scenario with 500 organizations was simulated on 30 servers and was used to estimate real-life performance of each of the systems.

Business Rationale and Choice of Metrics

To realize a vision of international adoption at scale, the overall infrastructure requirements of so many countries demand that blockchain architectures are interoperable at the data layer[1], but more important, that for specific use cases, these architectures can perform, scale, and consume as little power as possible.

Our philosophy is that the tokenization of multiple asset classes - money, documents/data, and non-cash assets - should be linked with the underlying principle of sovereign identity:

- Distributed assets of different types can be settled against each other more effectively using blockchain (delivery vs payment) if both assets are tokenized and linked to identity, e.g., to purchase with fiat currency a product, or to provide an incentive payment for an action
- Storage of high-value documents should be "on-chain" so that there is a shared and undisputable version of truth that the document conveys, whether it be a multi-party contract, credentials, IP rights, certifications, and many other pieces of information necessary to protect and trust in society.

Performance Metrics. We chose metrics that measure both financial transaction and document management performance. DLT Scalability and performance was measured using different KPIs to show all High Volume, High Velocity ("V&V") capabilities of the solutions, namely:

- transactivity - will be shown while performing eMoney transactions. The main KPIs are transaction per day and tps (transactions per second)
- data capacity - will be shown while publishing documents on DLT. The main KPI is number of published documents per day assuming that each document is 200 kB

Scalability Metrics. The question of scalability is whether adding more computational power will result in the capability to satisfy growing demand. As mentioned earlier, the systems were tested with 100 nodes (Early Adoption Scenario of 10 countries and 10 users), and 500 nodes (Larger Adoption Scenario). We note that multiple nodes can co-exist on the same server, and so while more nodes are being added, the network requirements are not such that there needs to be 5x more servers.

Power Consumption Metric. We chose to measure Watt-Hour per unit, i.e., either per document or per financial transaction. This is a universal metric and member states can use such a metric in financial models to understand economic benefits.

[1] Interoperability is not a topic of this effort, though it is noteworthy that the concern that blockchains are not interoperable is diminishing as it becomes more apparent that there are multiple ways to move data between chains. See: https://www.thestreet.com/technology/everything-you-need-to-know-about-blockchain-interoperability

Main Conclusions of Test Results

Overall, we observed that the Unified Enterprise DLT outperformed both of the Ethereum BESU consensus configurations (PoA-Clique and IBFT2) on (a) power consumption, and (b) performance metrics - capacity (for documents) or throughput (for monetary transactions).

More importantly, scalability was proven to be poor for Ethereum, and high for the Unified architecture.

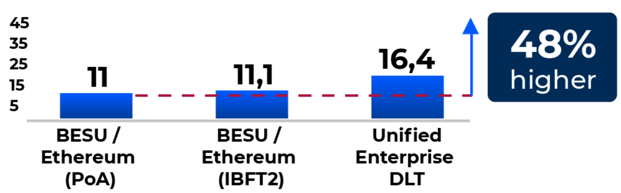
On-Chain Document Management Results:

“On Chain” documents are where the full document asset is sharded and written to the DLT, which is specifically of value where a document and its attributes should persist over time, e.g., contracts, property deeds, credentials, and authenticity verifications. This approach contrasts with the approach of writing a check-sum of a document to a hash, rather than the document itself.

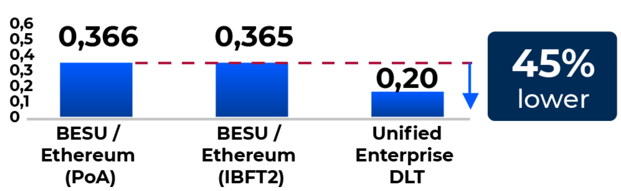
EXHIBIT 1
DOCUMENT MANAGEMENT TEST RESULTS

100 Participant Scenario
(10 in each of 10 countries)

Peak Load
mln documents / day

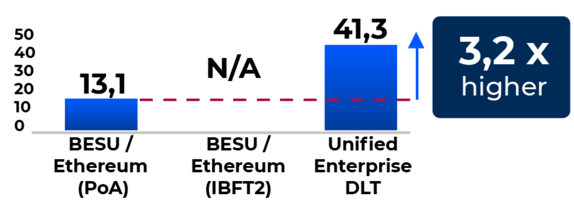


Power Consumption
mWh / document

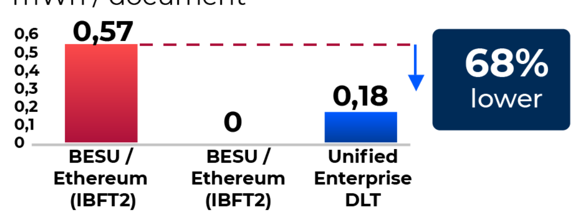


500 Participant Scenario
(25 in each of 20 countries)

Peak Load
mln documents / day



Power Consumption
mWh / document



Main points:

- For 100 participants (Early Adopter Scenario): the Unified Enterprise DLT outperformed the two Ethereum architectures, with 48% higher peak capacity and 45% lower power consumption.
- For 500 participants (Larger Adoption Scenario): the Unified Enterprise DLT outperformed the two Ethereum architectures with 320% higher (3.2x) peak capacity and 68% lower power consumption.

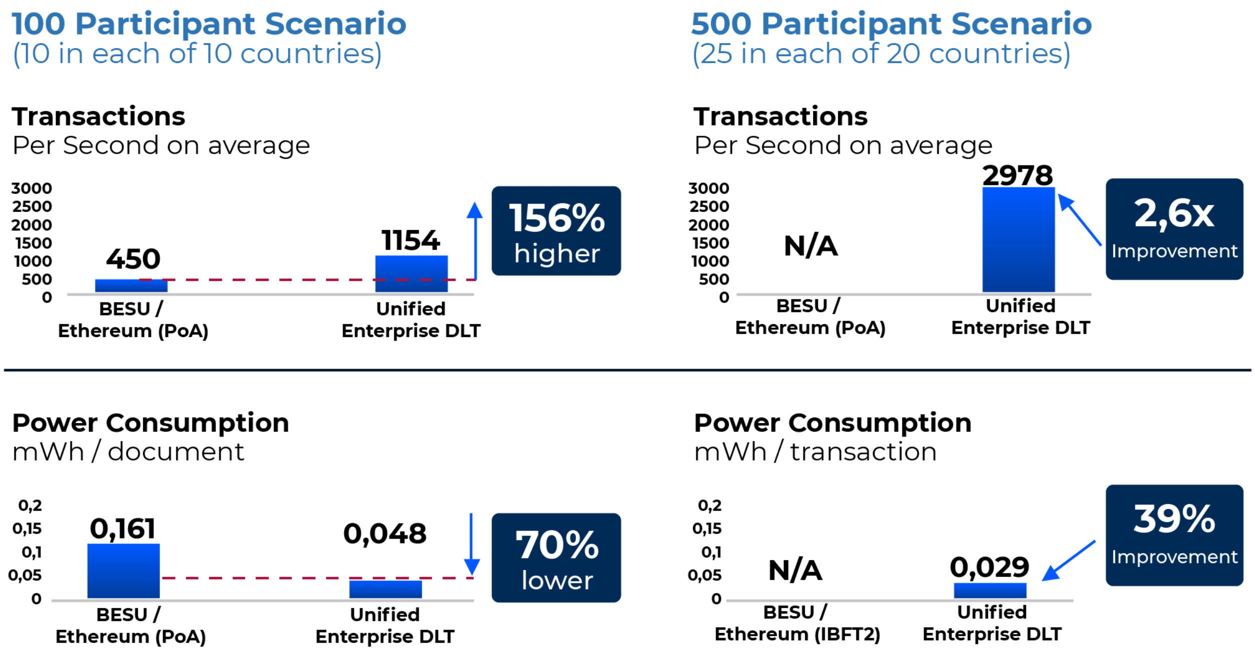
A point here about scalability. We note that with 5x more nodes, i.e., moving from 100 to 500 organizations,

- Ethereum's peak load increased by 19% (+2.1 to 13.1 from 11.0), and the Unified DLT increased by 2.51x.
- Ethereum's power consumption did not scale.   Instead, power consumption for Ethereum increased by 56%, a negative variant. For the Unified architecture, power consumption decreased by 38% , a positive result.

Financial Atomic Transaction Results

Financial transactions are much smaller in size than documents, and perform higher as would be expected. Effort was to make a true apples-to-apples comparison, using a chaincode version of Ethereum's ERC20 token standard vs Billon's use of a digital cash token.

EXHIBIT2
FINANCIAL ATOMIC TRANSACTIONS TEST RESULTS



We note that BESU tests at 500 nodes generated memory consumption bottlenecks, and it is possible with more RAM or specific optimisations, different results might be achieved. In order to maintain the integrity of the tests, all environments would need to use the changed configuration, and so we defer those tests to the next testing opportunity..

Main points:

- For 100 participants (Early Adopter Scenario): the Unified Enterprise DLT outperformed the two Ethereum architectures, with about 2.6x higher throughput and 59% lower power consumption

- For 500 participants (Larger Adoption Scenario): the Unified Enterprise DLT performed better than it did for 100 participants, but due to some memory bottlenecks with BESU/Ethereum, the comparison is not available at the time of the publication of this report.

A point here about scalability. For the Unified Enterprise DLT, the transaction throughput increased by 2.6x when adding 5x more nodes, increasing to close to 3,000 transactions per second. Power consumption improved by 40% as well. We note that financial transactions can be "batched", as many blockchains do in order to improve their throughput, but for these tests each blockchain transaction executed only 1 payment instruction of each organization.

Details of tests results

Early Adoption Scenario - On-chain Trusted Data Publication:

No of documents published per day	Unified Enterprise DLT	BESU/ Ethereum (PoA)	BESU/ Ethereum (IBFT2)
Maximum load ↑	↗ 16,4 mln docs/day	11,1 mln docs/day	11,0 mln docs/day
↓ Energy used, raw efficiency (miliWattHour)	↘ 0,20 mWh / document	0,365 mWh / document	0,366 mWh / document
Heavy production load ↑	11,0 mln docs/day	7,9 mln docs/day	5,9 mln docs/day
↓ Energy used, raw efficiency (miliWattHour)	0,43 mWh / document	0,448 mWh / document	0,517 mWh / document
Light load	1,9 mln docs/day	1,6 mln docs/day	1,5 mln docs/day
↓ Energy used, raw efficiency (miliWattHour)	1,47 mWh / document	1,649 mWh / document	1,65 mWh / document

The peak performance of Billon's Unified DLT is 48% higher than BESU/Ethereum, and the Unified DLT leads the efficiency with raw power consumption of only 0,20 milliWatt-Hour per document. There is negligible difference between BESU-PoA and BESU-IBFT2.

Early Adoption Scenario - Atomic Financial Transaction:

No of transactions processed per day	Unified Enterprise DLT	BESU/Ethereum (PoA)	BESU/Ethereum (IBFT2)
Maximum load ↑	↗ 99,7 mln tx/day (1154 tps)	N/A	22 - 43 mln tx/day (250 - 500 tps)
↓ Energy used, raw efficiency	↘ 0,0476 mWh/transaction	N/A	0,079-0,161 mWh/transaction
Heavy production load ↑	55,2 mln tx/day (639 tps)	N/A	17,6 mln tx/day (204 tps)
↓ Energy used, raw efficiency	0,0734 mWh / transaction	N/A	0,187 mWh / transaction
Light load ↑	27,2 mln tx/day (315 tps)	N/A	9,6 mln tx/day (111 tps)
↓ Energy used, raw efficiency	0,11 mWh / transaction	N/A	0,266 mWh / transaction

The peak performance of Billon's Unified DLT is significantly higher than BESU/Ethereum, however due to resource bottlenecks from JAVA VMs interrupting the tests, we can only estimate from partial results that the maximum performance of BESU-IBFT on this hardware environment would be in the range of 250-500tps based on the estimate of max possible CPU utilization and ideal block occupancy.

Larger Adoption Scenario - On-chain Trusted Data Publication:

No of documents published per day	Unified Enterprise DLT	BESU/Ethereum (PoA)	BESU/Ethereum (IBFT2)
Maximum load ↑↑	↗ 41,3 mln docs/day	N/A	13,1 mln docs/day
↓↓ Energy used, raw efficiency	↘ 0,18 mWh / document	N/A	0,57 mWh / document
Heavy production load ↑	19,5 mln docs/day	N/A	6,5 mln docs/day
↓ Energy used, raw efficiency	0,312 mWh / document	N/A	0,936 mWh / document
Light load ↑	3,9 mln docs/day	N/A	1,3 mln docs/day
↓ Energy used, raw efficiency	1,28 mWh / document	N/A	3,83 mWh / document

With Larger Adoption scale the maximum performance of Billon's Unified DLT has more than doubled as compared to the Early Adoption increasing throughput by +152%; while the peak

performance of BESU-Ethereum has shown scaling saturation with only +18% performance increase as compared to the smaller Early Adoption on-chain Publications.
At large scale and at peak performance, the advantage of Unified Enterprise DLT grows, with the energy efficiency being 3,1x higher than BESU at Larger Adoption scale.

Larger Adoption Scenario - Atomic Financial Transaction:

No of transactions processed per day	Unified Enterprise DLT	BESU/Ethereum (PoA)	BESU/Ethereum (IBFT2)
Maximum load ↑↑	↗ 257,3 mln tx/day (2978 tps)	N/A	N/A
↓↓ Energy used, raw efficiency	↘ 0,029 mWh / transaction	N/A	N/A
Heavy production load ↑	143,0 mln tx/day (1655 tps)	N/A	N/A
↓ Energy used, raw efficiency	0,043 mWh / transaction	N/A	N/A
Light load ↑	26,5 mln tx/day (307 tps)	N/A	N/A
↓ Energy used, raw efficiency	0,188 mWh / transaction	N/A	N/A

With Larger Adoption scale the maximum performance of Unified DLT's Atomic Financial Transactions has improved by 2,6x as compared to the Early Adoption e.g. increasing throughput by +158%, while improving the energy efficiency by further +39%.

Note: during BESU tests high memory consumption was presenting bottlenecks for some scenarios, it is possible that with much more RAM, or specific optimisations, different results may be achieved. For the Larger Adoption scenario we were unable to perform full duration test runs with 500 organizations for BESU-IBFT2, therefore the test was scaled down and performed with 400 organizations.

- **Maximum load -** this performance measure shows the push-to-the-limit DLT throughput that can be achieved on a given infrastructure; the limiting factor can be any bottleneck either CPU, network or disk subsystem hardware and shows how well optimised are the algorithms of the software L1 blockchain layer
- **Heavy production load -** this performance shows when a system operates at high business levels (typically targeted at 45% - 60% of maximum possible load)
- **Light load -** typical long term average of day-and-night level of transactions (5-15% of heavy production load)

Environment setup

For HL BESU ver 21.7.4 deployment setup:

- 1 peer node per organization
- 1 client application per organization
- 1 validating node per country (e.g. 10 or 30)
- Block creation: 2 sek (default settings)
- Maximum block size: default
- Ether: Increased gas consumption limit by factor of 1000x over typical cost of a contract, and wherever possible set free gas to unlimited value
- Two consensus protocols tested: IBFT 2.0 and Clique Proof-of-Authority
- Database chosen: RocksDB

For Billon's Unified DLT ver 3.18 deployment setup:

- 1 pnode: publication node per organization
- 5 or 10 cnodes: corporate maintenance nodes per organization
- 1 mnode: minting nodes per country (e.g. 10 or 25) for governmental currency, assets or token creation
- 1 certificate authority node per country
- Diagnostic logging turned-off and code compiled with maximum optimizations and assembly for vectorised cryptography, all the other settings and configuration of the nodes are in default configuration
- Database chosen: BDB

The hardware environment consists of 10 servers for Early Adoption Scenario and 30 servers for Larger Adoption Scenario with the following specifications:

Early Adoption Scenario:
10 x Dell PowerEdge R6515:

- 1x AMD EPYC™ 7502P 32-Core
- 128 GB DDR4 ECC
- 2x 1.92 TB NVMe SSD (Samsung)
- RAID controller Dell PERC H730P
- 1 GBit/s-Port
- 2x PSU: Redundant Platinum Certified Hot Plug

Larger Adoption Scenario:
30 x AMD Ryzen:

- 1x AMD Ryzen 9 3900 12-Core
- 128 GB DDR4 ECC
- 2x 1.92 TB NVMe SSD (Samsung)
- 1 GBit/s-Port
- 1x power supply

All of the nodes are evenly distributed among all of the physical servers, for efficiency running in shared user space e.g. without any virtualization or isolation layer. The servers are connected by a 1Gb bandwidth LAN network without any VPN or any firewalls between servers.

On every server a standard POSIX OS is installed: Linux Ubuntu 20.04 LTS, with the following modifications:

- apt install python3 python3-yaml zstd p7zip-full gdb screen rsync binutils
- apt install zabbix-server-pgsql zabbix-frontend-php php7.4-pgsql zabbix-apache-conf zabbix-sql-scripts zabbix-agent
- /etc/sysctl.d/10-ptrace.conf -> kernel.yama.ptrace_scope = 0
- /etc/security/limits.conf -> kolonia soft nofile 10240
- /etc/security/limits.conf -> kolonia hard nofile 10240
- systemctl enable zabbix-server zabbix-agent apache2
- iptables:
    - -A INPUT -i lo -m comment --comment "Allow all incoming on iface Loopback" -j ACCEPT
    - -A INPUT -p icmp -m comment --comment "Allow incoming PING" -j ACCEPT
    - -A INPUT -m state --state RELATED,ESTABLISHED -m comment --comment "Accept RELATED,ESTABLISHED packets" -j ACCEPT
    - -A INPUT -m conntrack --ctstate INVALID -m comment --comment "Drop all invalid packets" -j DROP
    - -A INPUT -s x.x.x.x/32 -m comment --comment "Allow traffic to all colony servers" -j ACCEPT
    - -A INPUT -j DROP
    - -A OUTPUT -j ACCEPT

Contracts description:

For **Atomic Financial Transactions** a chaincode version of Ethereum's ERC20 token standard (based on OpenZeppelin's implementation) is used, modified only to accept SOAP calls from the test script harness. ERC20 standard is widely used to implement both non fungible assets or versions of stable tokens. For Billon's Unified DLT we use it's native implementation of Distributed Digital Cash which provides FIAT governmental currency implementation of technology compliant with EU directives and regulations, which also can be used to implement various non-fungible tokens and assets. Each ERC20 contract associated with one organization is used to perform atomic transactions transferring value to another randomly chosen wallet, in parallel with all the other organizations.

Trusted Documents On-chain Publication: since shared business data often represents substantial value and moreover data related to end customers is subject to regulatory oversight, it is important to ensure proper protection of such data. Ideal protection can be provided by storing such valuable, trusted data directly on the ledger e.g. "on-chain". Therefore a custom smart contract is created containing a single BLOB with randomly generated business data (binary large object) field and an accompanying map of metadata fields. No encryption of on-chain data or access control is attempted as it is not supported by HL/BESU and only partially by HL/FABRIC.

The Smart Contract is simple, including only: blob, id, and a map of metadata describing published documents, all represented as strings. There are two methods to put and query the document on a blockchain. For the end of document publishing we will consider the time of committing the data block to peers. The contract should be endorsed by the majority of organizations. Each organization publishes independently on its own on-chain contract. The size of on-chain trusted data was set to 200KB for each publication.

Test methodology

We run BESU/Ethereum in two configurations: IBFT and PoA, for a total of 24 test runs. For Billon's Unified DLT we run 12 test runs each, grouped in the following deployment scenarios: Early Adoption On-Chain Documents, Larger Adoption On-Chain Documents, Early Adoption Atomic Transaction and finally Larger Adoption Atomic Transactions; each in three transaction load steps. The same hardware is used for all of the sixteen deployment scenarios, with each test scenario running for approximately 1 hour with changing transaction load put onto the DLT network by the Python test harness. The test load is evenly spread across the nodes of all of the organizations by the test script. Between the DLT environment tests switchover, all of the data is being deleted and servers restored to their original condition. 24 of the test runs were performed on a 10-server hardware environment (Early Adoption), and the other 24 test runs were performed on a 30-server environment (Larger Adoption).

During each of the deployment tests, we modify the transaction load in a stepwise fashion, in three load-level steps. The steps are calibrated such that on the highest step all three DLTs will be allowed to report failed transactions, however the best performing DLT shall perform at least >=99% of successful transactions. On lower lower steps, the best performing DLT has to achieve a success rate of >99.99%. With the transaction workload on the lowest step, all three systems are expected to successfully complete the submitted transactions. The lowest step is approximately calibrated to be about 5 - 10% of maximum load. The intermediate step is set at aproximatley 45% - 55% of the max, which represents typical sustainable production load in any hardware setup. We analyze power consumption changes related to lower transaction workload on each of the steps. Work submitted by the test harness for each step is identical for each of the DLTs. The test harness script for high performance on-chain trusted data publishing has built-in error monitoring and a parameter for minimum and maximum generated load on each step. The minimum generated load is set up to be half of the target load for each of the steps. If the DLT nodes respond with errors or do not respond with job completed status within the specified timeout period, the test harness gradually throttles down the generated load. If the waiting jobs queue decreases, the test harness gradually throttles up the load. The automatic load adjustment stops at the specified minimum and maximum thresholds. It will be up to the network nodes to deal with possible overload if the minimum specified target load for a given step exceeds the processing capacity of the network.

During all Early Adoption Scenario tests hardware power monitoring is performed on the servers, measuring power consumption with 10 second time resolution. During each blockchain test, the best performing load step will be selected for each of the DLTs, and all of the successfully completed transactions during the selected step will be summed-up for a total number of transactions performed. Next, power consumed by all of the servers during the time needed for the selected step will be summed-up for a total power consumption of the simulated system. Finally the total number of transactions will be divided by total power consumed to calculate the power efficiency of each of the DLTs.

During the Larger Deployment Scenario we concentrate on absolute throughput numbers and performance of each of the DLTs. This scenario demonstrates the path to large scale adoption of blockchain technology and any potential performance gaps that still will need to be worked on. The hardware power consumption is only estimated from typical power consumption of these servers under full load. The power efficiency numbers from this scenario are less precise but they do help to validate overall efficiency metrics.

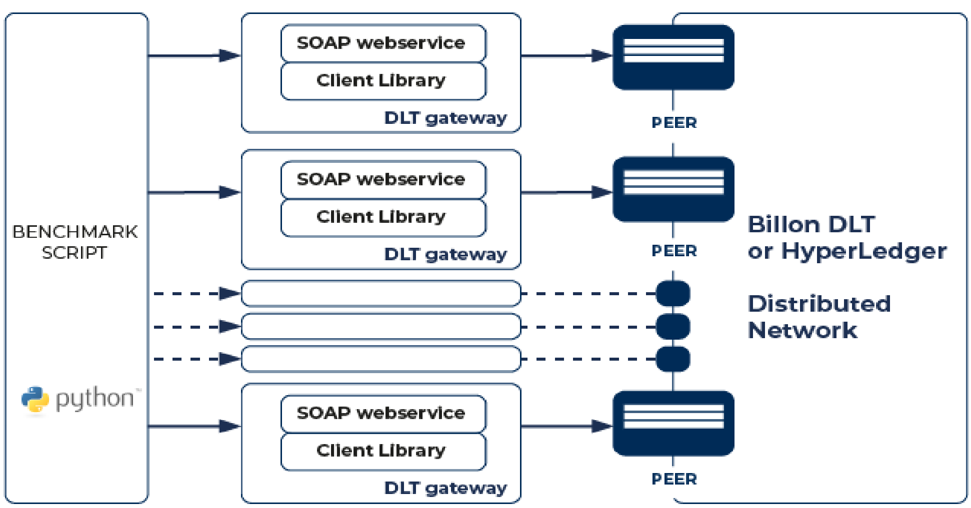
Test setup

The objective of the test is to measure performance in two business scenarios: publication of large data objects (for instance documents or attestation data) directly on-chain; and second scenario: p2p transactionality of monetary values or non-fungible-tokens (NFTs).

- **Trusted Documents and Business Data On-Chain Publication**
  Since only Billon's Unified DLT technology supports on-chain publication of private GDPR-compliant data directly on the ledger, for comparative purposes benchmarking was performed on public documents, which all three DLTs to various extent support.
  The size of the published data (public documents) will be set to 200KB for a single document size. The document will be non-compressible randomly generated data.

- **Atomic Financial Transactions**
  The scenario is designed to simulate the needs of financial institutions or individuals wishing to directly transact p2p value transfers without the intermediation of a trusted 3rd party. Since only Billon's Unified DLT supports e-money FIAT governmental currencies, the Hyperledger smart contracts will transact NFT assets (ERC20) for benchmarking purposes.

There will be one instance of a simple benchmark script, written in Python, that will communicate with client nodes written in Java using SOAP API (the same API is exposed in our publishing nodes in Billon which we believe are equivalent to Hyperledger client nodes). The script will manage the number of published documents, publishing rate and will be responsible for load balancing.
The test harness consisting of a Python benchmark script and network deployment scripts are running on a separate machine so as not to interfere with performance of DLT network machines.

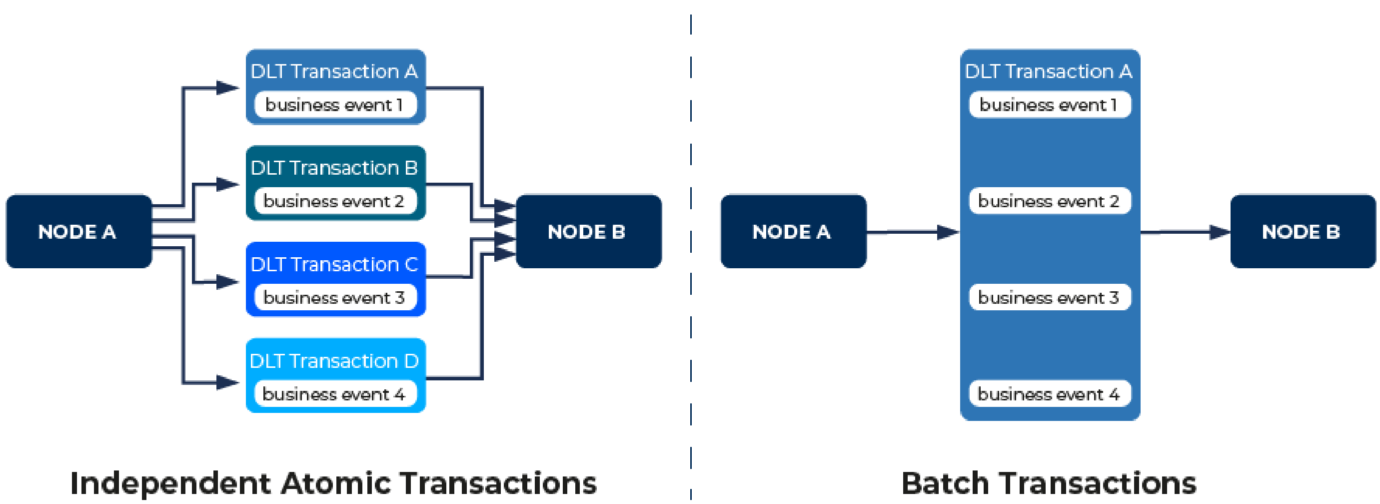


Atomic transaction vs Batching transaction

Billon Unified Enterprise blockchain ensures that high volume and throughput is reached on the atomic transaction level. This means that a single transaction holds an indivisible and irreducible series of information and is always performed with guaranteed coherency between any other similarly guaranteed transactions. This means that a single business event is equal to a single DLT event. This allows that all of the transactions are independent from each other, and a problem with one business transaction does not impact any other transactions. Atomic transaction throughput is the best measure to compare different systems as it compares similar types of operations and does not leave any substantial room for interpretation of the results.

Batching transactions means that multiple business events are collapsed into a single DLT transaction. From a business perspective, these aggregated DLT transactions can be divided into smaller business objects by the off-chain business logic. The biggest challenge of that approach is that as business events are not treated independently from each other (at least from system perspective), they are in fact forced to be artificially hard-linked on the Distributed Ledger. In case there are any problems with a single business event, it forces an invalidation and costly exception handling to all of the events that are encapsulated into a single Blockchain transaction. Additionally, batching transactions makes comparison between different systems difficult to interpret.

Billon's Unified DLT system can batch transactions with good, proportional/linear impact on performance (we estimate that batching 10 business events increases apparent business throughput by 9 times). For the purpose of performance measurement to ensure the comparability of the solution the batching is not used.



Energy Efficiency Analysis

None of the three tested blockchain systems uses Bitcoins' original proof-of-work which has been a hugely energy intensive process to achieve guaranteed data coherence. Nonetheless, there remain significant differences in advancement of DLT technology and the impact they have on the power efficiency varies with deployment type. The benchmarking efficiency analysis was designed to accurately capture the differences and assess whether the new DLT technology advancements are sufficient to be a real alternative to current legacy centralised IT systems.

Our benchmarking test setup is fully self-contained, meaning it contains all of the types of nodes and all of the DLT system's components are hosted on Dell Servers and power consumption is added up from all of them, from the commencement of the test until the test completion. Power consumption is then expressed in total Wh consumed by all of the servers. The power consumption measurement is done directly by servers' motherboards. The benchmarking test harness resides on a single separate server and its power consumption is excluded from the total. In case of Billon's Unified DLT the peer nodes are light enough to be run directly on the end-user smartphone device in a fully peer-to-peer mode with a smartphone or tablet of a merchant, therefore no additional hardware or merchant terminals are required.

From each of the test runs, for all three DLTs, we take the best performing step and count the total number of transactions or documents published during the 15-minute long step. (For BESU/Ethereum we take two best performing steps, one for PoA and one for IBFT configuration.) We then divide total power consumed during the test by total number of transactions (or documents) processed during the test to obtain a measure of every consumption of a single operation, for ease of presentation we use units of miliwatts (1/1000 Watt Hour). e.g. mWh / atomic financial transaction, and mWh / trusted document secured.
Since running the servers to capacity, in a common and shared cloud environment, presents the unrealistically best case scenario, we need to make two extrapolation adjustments: (1) level of hardware multi-tenancy e.g. how many physical servers would be shared between different organizations (either via virtualization or via cloud); (2) long term average utilization hardware ratio, e.g. most of the time servers have to be ready to accept maximum workload, rather than actually process it.

In order to cope with the high load that arises on days such as Black Friday Sale, the infrastructure that supports such scenarios must have sufficient capacity to cope with it. A system failure during such a short time event inevitably leads to a high loss of confidence and damage to the reputation of the technology involved.
See:https://cloud.google.com/architecture/black-friday-production-readiness
"Increase in traffic from 5x to 20x or greater, with generally higher conversion rates and greater loads on backend systems."
See:https://www.finextra.com/blogposting/20963/european-blockchain-services-infrastructure-ebsi-the-european-way-to-get-most-out-of-blockchain
Therefore for the extrapolated real life energy consumption we need to use **Light Load** raw efficiency energy numbers, rather than Maximum Load numbers, and then we further need to use 2x adjustment to power consumption to reflect the ability to process peak loads of 20x load.

Hardware multi-tenancy e.g. cloud or co-shared virtualization is still in the early adoption phase in traditional industries. In our tests we presented a multi-tenancy factor of 16x for Larger Adoption Scenario, and multi-tenancy of 10x for Early Adoption Scenario (or full cloud adoption). However since multi-tenancy is still in its early stages, we make conservative assumptions that it will not be used at all, resulting in a 10x factor of extrapolation adjustment.

We also make the availability and up-time adjustment, e.g. each organization will run a double set of nodes to ensure High-Availability Clustering (e.g. 2 need for spread load to two physical servers) and that results in a 2x factor of adjustment.

Finally the data center energy consumption of servers represents only a portion of the total power consumption, because of the cooling requirement as well as necessary secondary systems, lighting security, etc therefore fully loaded power consumption is additional 40-80% power usage.

The combined factor for the energy extrapolation used in the real life estimation is therefore 40x to 70x, with the likely mean of 50x.

Real Life Estimation of Energy Efficiency:

	Billon's Unified DLT	BESU/Ethereum (PoA)	BESU/Ethereum (IBFT2)
Early Adoption Documents (raw efficiency)	1.47 mWh / document	1.649 mWh / document	1.65 mWh / document
On-chain Trusted Data Publication **Energy Efficiency** (real life extrapolation)	**0.06 - 0.10 watt hour / document**	**0.066 - 0.12 watt hour / document**	**0.066 - 0.12 watt hour / document**
Early Adoption Atomic Financial Transaction (raw efficiency)	0,0476 mWh/ transaction	N/A	0,079-0,155 mWh/ transaction
Atomic Financial Transaction **Energy Efficiency** (real life extrapolation)	**0.02 - 0.033 watt hour / transaction**	N/A	**0.03 - 0.11 watt hour / transaction**
Combined Trusted Document & Financial Transaction Efficiency	0.08 - 0.133 watthour / combined	N/A	0.10 - 0.23 watthour / combined

For the full scale, worldwide system-transitioning of all major payment schemes' infrastructure of credit and debit cards that would switch from centralised data rooms to adopting Unified DLT, the overall power consumption would be reduced by 90% - 95%. That translates into proportional benefit to the climate, and at the 2020 levels estimated at 412 thousand tonnes of CO_2, it has the potential of lowering total emissions by 350 to 390 thousand tonnes of CO_2 per year.

Measurements and test data gathering

All of the hardware monitoring is performed by the Zabbix tool (https://www.zabbix.com/) which collects data directly from PC sensors on Dell Servers for total power consumption and from the Linux kernel measuring CPU consumption, network traffic intensity and disk write and read rates. The data is collected in 10 second intervals, and then upon test completion exported to a .csv format file for each server separately. The following values are monitored for each of the servers: Power Supply usage, System load, CPU utilization, Detailed CPU usage, System load, Memory Usage, Disk average waiting time, Disk read/write rates, Disk utilization queue, and Network traffic.
Sample CSV export is attached in Appendix 2

For the purposes of the Larger Scale Adoption scenario, we have used AMD-based servers with ASUS Pro WS 565-ACE motherboard with power efficient Ryzen 3900 processor, 128GB ECC RAM and Samsung MZQL21T9HCJR NVMe disks. This particular motherboard does not have full power monitoring, therefore we rely on external wattometer measurements which report that at Idle state each server consumes ~65W, and under max load the power consumption rises to approx. ~165W per server.

The Python test harness is responsible for management of jobs dispatched to DLT network nodes and collecting the data with timestamps measuring, job initiation, job status and time of the job completion. All of the measurement data is exported in the CSV format for all the servers together with all the identifying details in each row of the exported data. On a 30-server environment two simultaneous instances of test script are used in order to mitigate python performance limitations, while on a 10-server environment a single instance of python test script is sufficient.

Sample CSV export from test harness is attached in Appendix 2

APPENDIX 1: Smart Contracts code used for deployment

The source code for test benchmarking and measurement, as well as smart contract and Java adapters is posted on the Github at the following URL:
https://github.com/Billongroup/bechmarking-DLT-energy-efficiency

a) BESU smart contract for On-Chain Documents:

```
pragma solidity >=0.7.0;

contract DocumentsStorage {

 struct DocumentInfo {
   bytes documentData;
   string documentMetadata;
 }

 mapping(bytes32 => DocumentInfo) publishedDocuments;
  event PublishedDocument(bytes32 documentHash, address publisher);

  function publishDocument(bytes memory documentData, string memory documentMetadata)
      public {
   bytes32 documentHash = sha256(abi.encodePacked(documentData));
       require(publishedDocuments[documentHash].documentData.length  ==  0,  "Document
      already exist");
   publishedDocuments[documentHash] = DocumentInfo(documentData, documentMetadata);
   emit PublishedDocument(documentHash, msg.sender);
 }

  function  readDocument(bytes32  documentHash)  public  view  returns  (DocumentInfo
      memory) {
    require(publishedDocuments[documentHash].documentData.length > 0, "Document doesnt
      exist");
   return publishedDocuments[documentHash];
 }
}
```

b) BESU smart contract for Atomic Financial Transaction:

Source code: https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/token/ERC20/ERC20.sol

```
pragma solidity ^0.8.0;

import "./ERC20.sol";

contract BIL is ERC20 {

   constructor() ERC20("BIL", "BIL") {
     _mint(msg.sender, 1000000000 * 10 ** uint(decimals()));
   }

}
```

APPENDIX 2: Sample reporting and measurement data

Atomic Financial Transaction - sample detailed report:

ID STATUS START_TIME END_TIME PAYER RECEIVER AMOUNT CURRENCY

```
8AcR84MWumnrSvD8LCrLCJUi FINISHED_OK "2021-10-12 18:16:20" "2021-10-12 18:16:22"
cnode3_97 cnode3_96 10 BIL

8Pfu8E4WAFCEcbWye4V8Eo1U FINISHED_OK "2021-10-12 18:16:20" "2021-10-12 18:16:22"
cnode3_97 cnode3_96 10 BIL

89kWVvLRSoxiGUr9oJYjShtY FINISHED_OK "2021-10-12 18:16:27" "2021-10-12 18:16:28"
cnode3_97 cnode3_96 10 BIL

86PoQ49dUR85YsxNNcCbsZam FINISHED_OK "2021-10-12 18:16:27" "2021-10-12 18:16:28"
cnode3_97 cnode3_96 10 BIL

869pMgBfuLtdqVsS6dFVjhTZ FINISHED_OK "2021-10-12 18:16:33" "2021-10-12 18:16:35"
cnode3_97 cnode3_96 10 BIL

8HmPjwAhtd3LPpMrfpCfScPt FINISHED_OK "2021-10-12 18:16:33" "2021-10-12 18:16:34"
cnode3_97 cnode3_96 10 BIL

89hoDEBbAE31vqLJf2wBF7Qs FINISHED_OK "2021-10-12 18:16:40" "2021-10-12 18:16:41"
cnode3_97 cnode3_96 10 BIL
```

(...)

On-chain Trusted Data Publication - sample detailed report:

```
DOC_HASH MD5 PUB_TASK_ID PUB_ADDRESS CIF DUR_TIME STATUS PUB_END_TIME THREADS
START_BRG_TIME CREATE_BRG_TIME PUBLISHED_BRG_TIME DUR_BRG_TIME

ZVo1BLpQn4cp845WwJRmwNW5fYWdYzgRBs5faQoT1iC2evosZAGsftyU959RShdrNaQTBrLX8eU7o5zAh71p9zRVEWC
yrTTBCsWbBMRhJwqkAUnvQ 984f410282a8086896c13f60e617dc07 CEJCdc84hgqHCnKnfd264ss8
http://116.202.250.24:17014 no_cif 185.297 FINISHED_OK 1634061361.214 37 1634061176.568
1634061177.034 1634061180.111 3.543

Z3wVxFyA75x3gegBj51i4wb2VQhFQ9N853PjbsiTHjc2ikHWNUX3qmoxjEGutNqcXZBEP
9ba64da0f737efb5d1d4f4a16aac39a8 CFerSAFwKpK9JJLJwFuDrDPz http://116.202.250.24:17014
no_cif 185.260 FINISHED_OK 1634061366.229 37 1634061181.039 1634061181.343 1634061184.226
3.187

Z4SPEeLYZZLw3gcAE16bsXu1NK4yKbHvki85aUkrcqNxAFruHEL2HYALRNSYxBEH2VX2hc3BjhdwHgNXsqW5SjxCCgu
3pw3SAMMN82SsqXaHnF2RN a54e418c557e2df131a55ccd89fd505d C1x8YuZB6qqNagYBXhPFpLB9
http://116.202.250.24:17014 no_cif 185.282 FINISHED_OK 1634061371.255 37 1634061186.160
1634061186.366 1634061188.492 2.332

ZSdAHN5Hri2ZY8UcYcJapKY1Bnswj9j6MxE8areh4hUB4jKeFdP1SJYwDSQrGsdSjt4bMDea5YcMiwadUeNptYVzyTp
86c9QWZAV7Yvm4PyJfyGgr a580cd25ed0ed310e1c91731da75a05a CYBqovG8AgQKqR6gQBg49zsN
http://116.202.250.24:17014 no_cif 185.297 FINISHED_OK 1634061376.274 37 1634061191.082
1634061191.506 1634061192.690 1.607

ZUVSgDy4uabD5osSjhyFCTK3Qg2QMQGaKYS88c1KkZRfD93EcZiCq7sK8vEVhjEkqqo8ZmqEDgWnvsiEK9bR9MLMJBP
smzk6DxNsgedTdBWBe9fn1 3c342ddb100a3c3d23c9b047f86da1f9 CFCmBhjQkTWfzgQ5TcoHE5xQ
http://116.202.250.24:17014 no_cif 185.310 FINISHED_OK 1634061381.291 37 1634061196.009
1634061196.220 1634061198.129 2.120
```

(...)

Power consumption and utilization - zabbix sample report:

Time	dellR6515-10: Instantaneous power reading
2021-10-11 21:30:00	118 W
2021-10-11 21:30:10	119 W
2021-10-11 21:30:20	122 W
2021-10-11 21:30:30	117 W
2021-10-11 21:30:40	120 W
2021-10-11 21:30:50	121 W
2021-10-11 21:31:00	120 W
2021-10-11 21:31:10	114 W
2021-10-11 21:31:20	118 W
2021-10-11 21:31:30	122 W
2021-10-11 21:31:40	117 W
2021-10-11 21:31:50	113 W

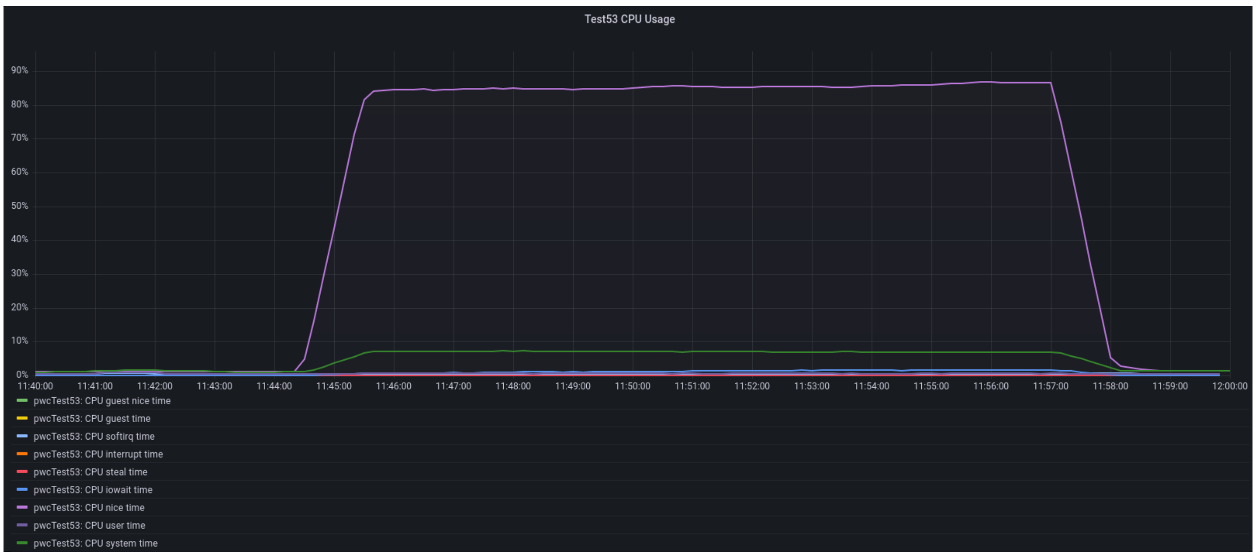
(...)

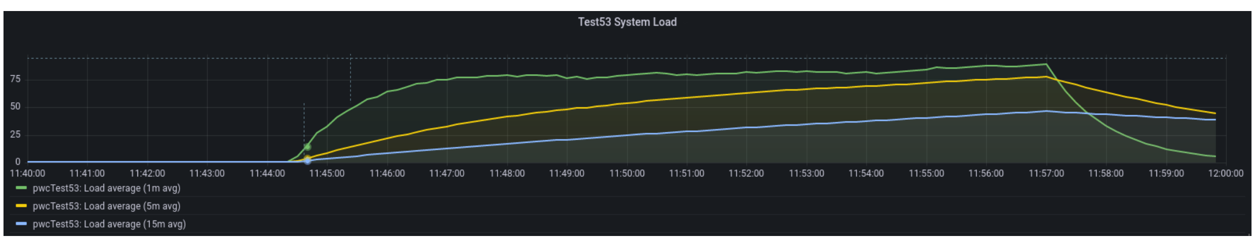
Time	dellR6515-10: Load average (1m avg)	dellR6515-10: Load average (5m avg)	dellR6515-10: Load average (15m avg)
2021-10-11 21:30:00	2.85	6.54	9.12
2021-10-11 21:30:10	2.49	6.36	8.98
2021-10-11 21:30:20	2.18	6.16	8.89
2021-10-11 21:30:30	2	6.03	8.85
2021-10-11 21:30:40	1.91	5.84	8.77
2021-10-11 21:30:50	1.69	5.7	8.65
2021-10-11 21:31:00	1.81	5.55	8.61
2021-10-11 21:31:10	1.53	5.38	8.52
2021-10-11 21:31:20	1.37	5.24	8.4
2021-10-11 21:31:30	1.47	5.13	8.36
2021-10-11 21:31:40	1.47	4.97	8.29

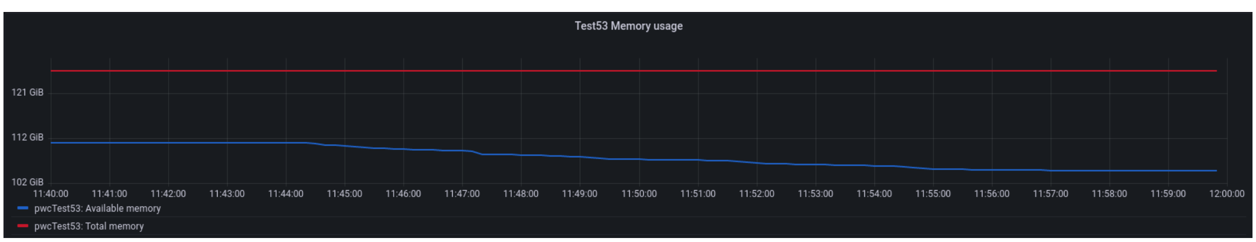
(...)

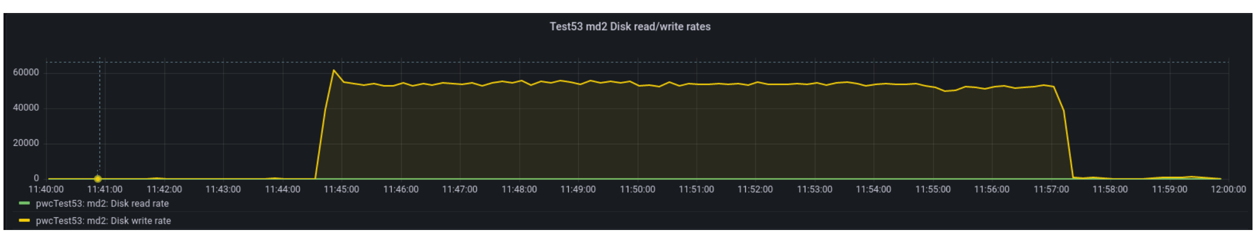
APPENDIX 3: Sample monitor of key hardware resources

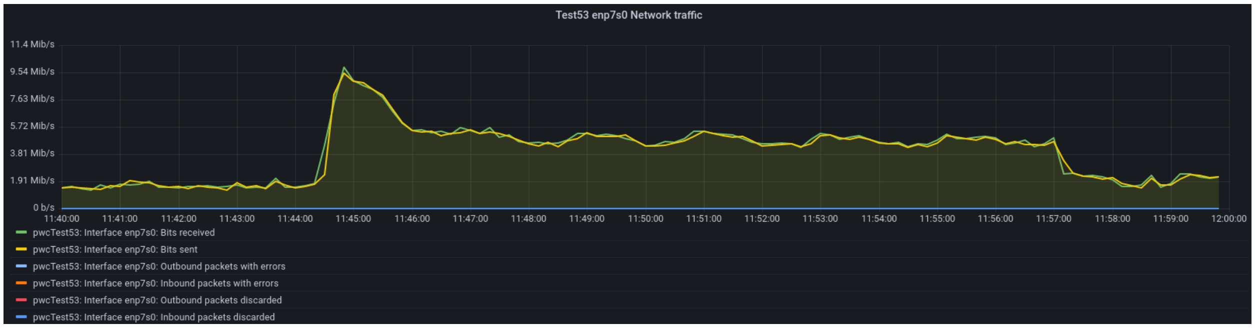
Early Adoption On-Chain Documents - Billon's Unified DLT:
Oct 11, 2021 @ 21:40 CET; Oct 12, 2021 @ 19:52 CET;
Early Adoption Atomic Financial Transactions - Billon's Unified DLT:
Oct 12, 2021 @ 17:08 CET; Oct 12, 2021 @ 17:35 CET;
Larger Adoption On-Chain Documents - Billon's Unified DLT:
Oct 16, 2021 @ 17:10 CET; Oct 16, 2021 @ 18:08 CET;
Larger Adoption Atomic Financial Transactions - Billon's Unified DLT:
Oct 15, 2021 @ 11:40 CET; Oct 15, 2021 @ 12:55 CET; Oct 15, 2021 @ 13:47 CET



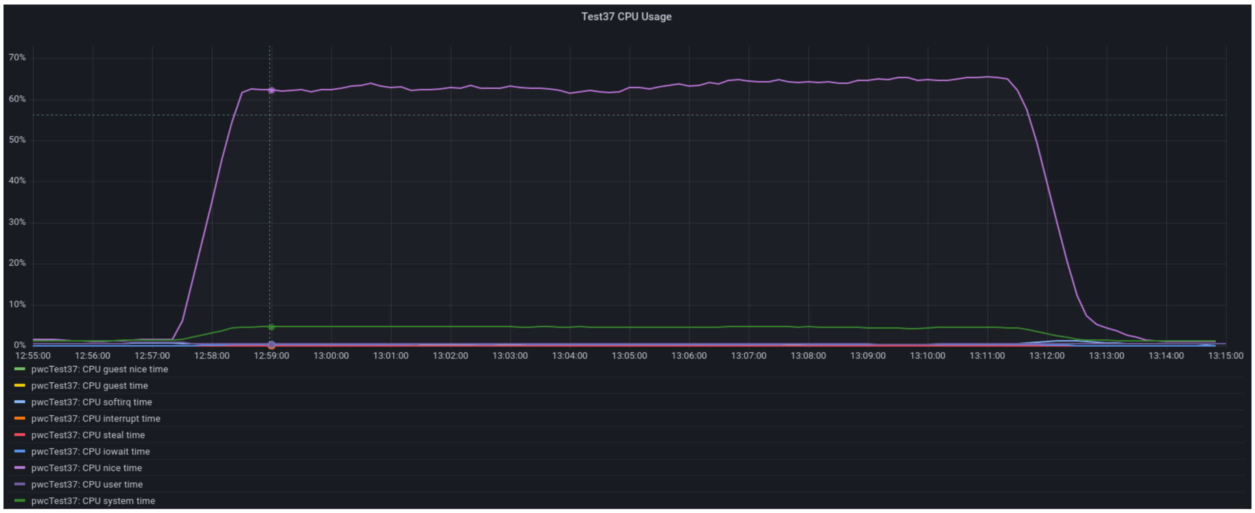


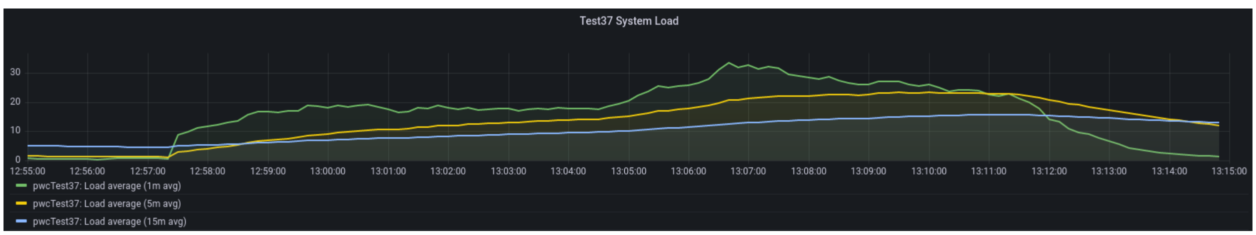


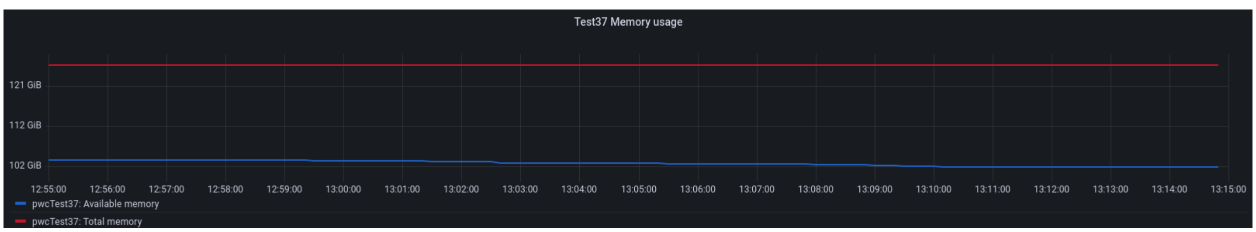




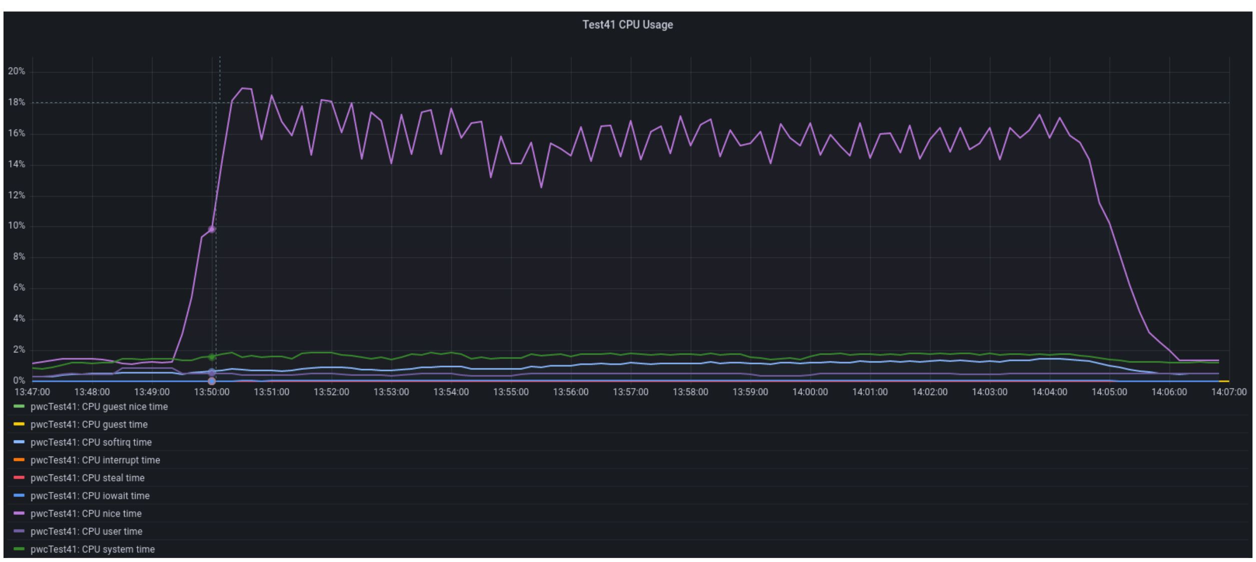
Oct 15, 2021 @ 12:55 CET;



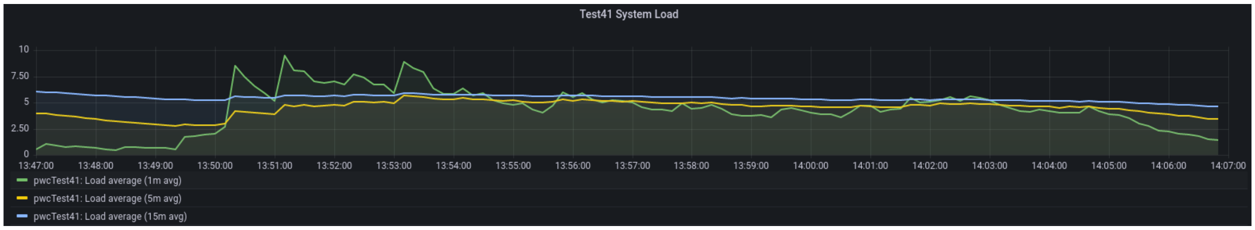


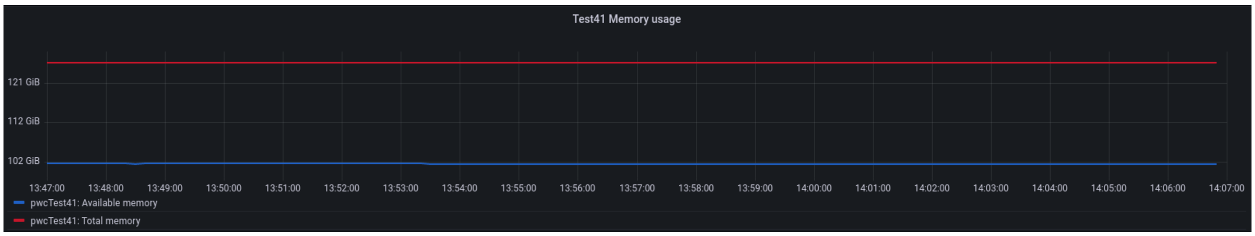


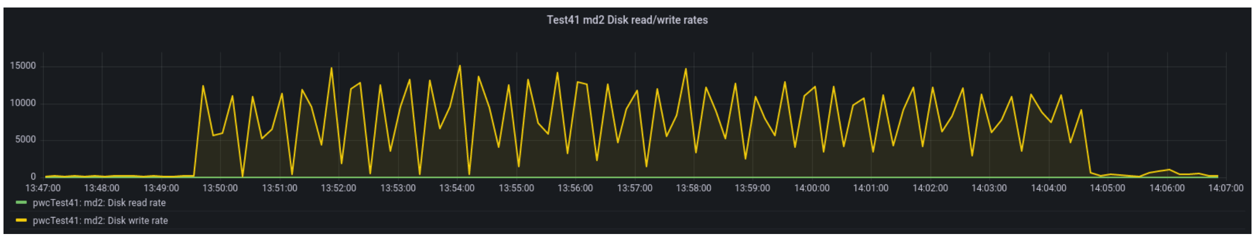
Unified Enterprise DLT Early Adoption, on-chain publications:

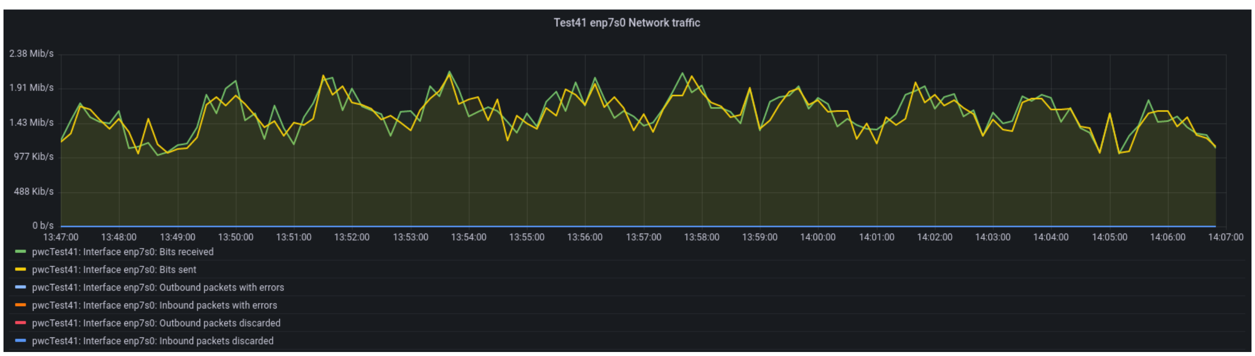


Oct 15, 2021 @ 13:47 CET

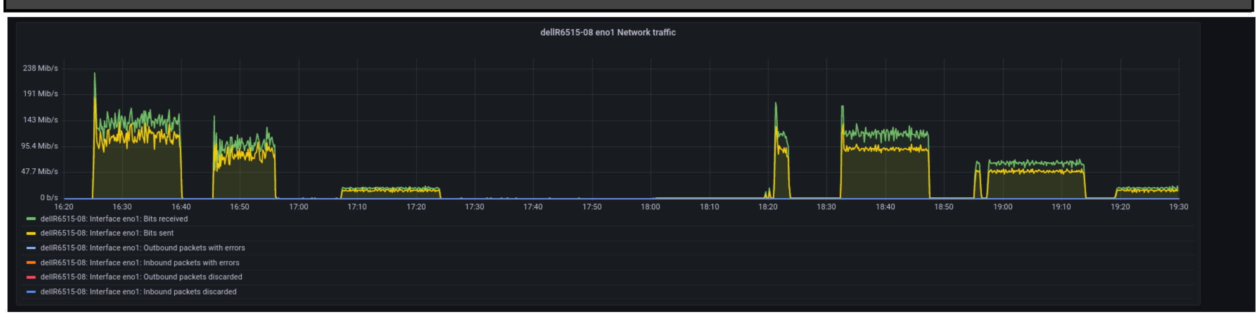


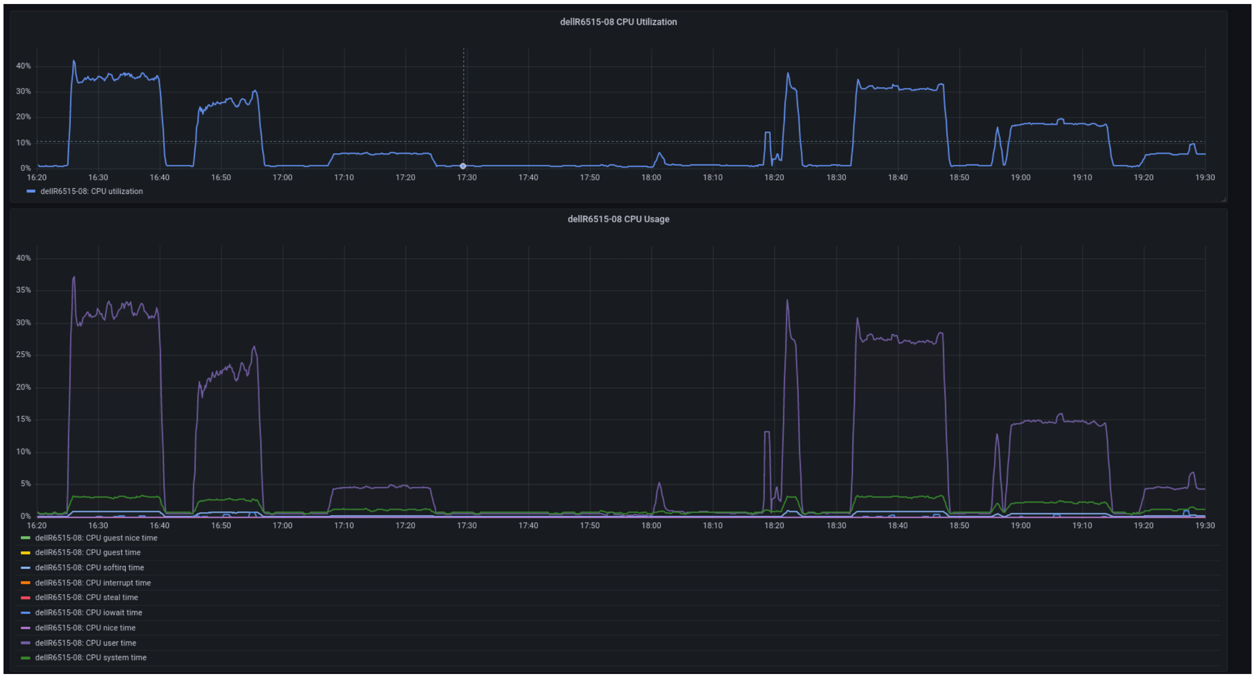


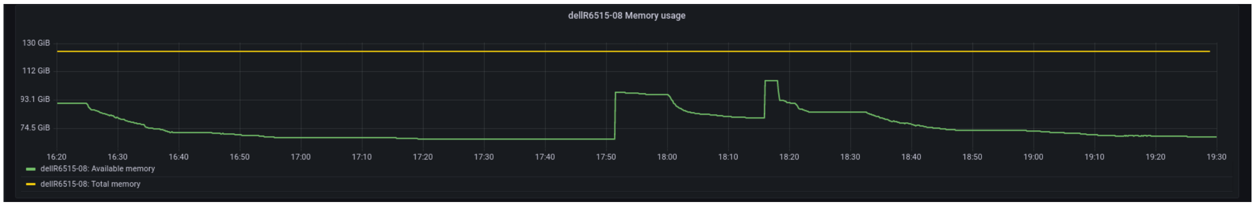


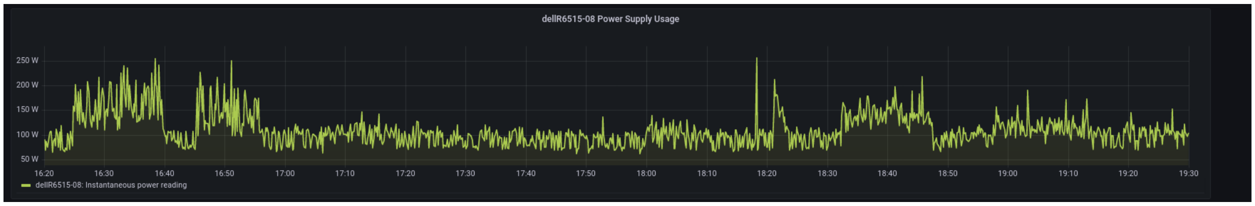


Oct 15, 2021 @ 16:40 CET - Oct 15, 2021 @ 19:30 CET;
Early Adoption Atomic Financial Transactions - Hyperledger BESU/Ethereum DLT:

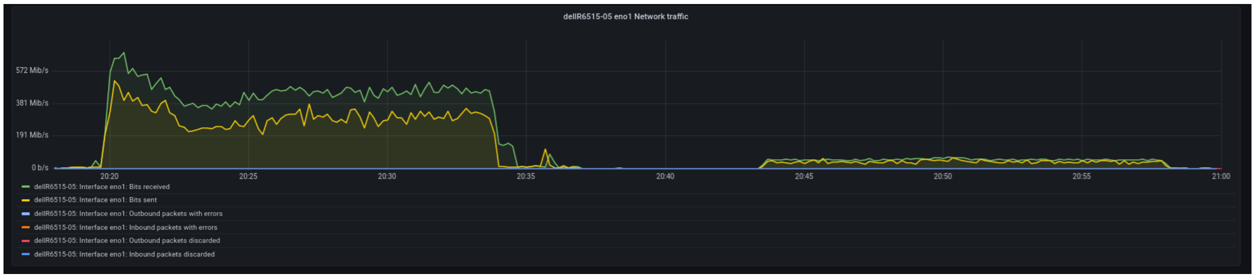


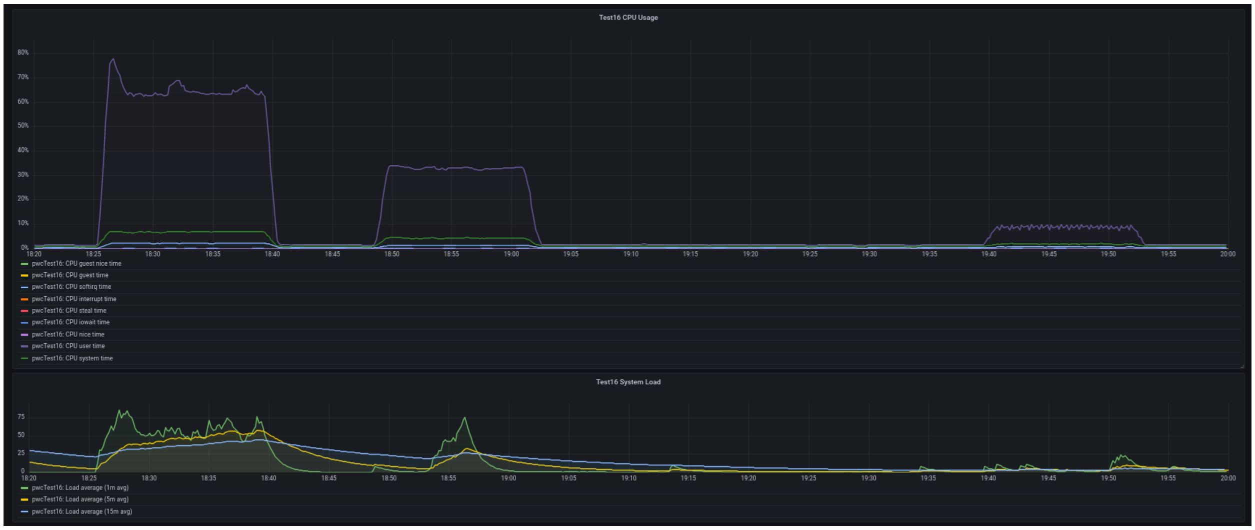






BESU Larger Adoption, on-chain publications:




Test16 CPU Usage
pecTest16: CPU guest nice time
pecTest16: CPU guest time
pecTest16: CPU softirq time
pecTest16: CPU interrupt time
pecTest16: CPU steal time
pecTest16: CPU iowait time
pecTest16: CPU nice time
pecTest16: CPU user time
pecTest16: CPU system time
Test16 System Load
pecTest16: Load average (1m avg)
pecTest16: Load average (5m avg)
pecTest16: Load average (15m avg)


Test16 Memory usage
pecTest16: Available memory
pecTest16: Total memory
Test16 md2 Disk read/write rates
pecTest16: md2: Disk read rate
pecTest16: md2: Disk write rate
Test16 enp7s0 Network traffic
pecTest16: Interface enp7s0: Bits received
pecTest16: Interface enp7s0: Bits sent
pecTest16: Interface enp7s0: Outbound packets with errors
pecTest16: Interface enp7s0: Inbound packets with errors
pecTest16: Interface enp7s0: Outbound packets discarded
pecTest16: Interface enp7s0: Inbound packets discarded

APPENDIX 4: PWC Sample Energy Efficiency Analysis

In [1]:

```python
import pandas as pd
from datetime import datetime, timedelta
import os
import matplotlib.pyplot as plt
```

loading the server HW monitoring data

Now the billion test files are loaded

In [2]:

```python
folder_path_logs = r'C:\temp\billion_data_sorted\retest_18\\'
```

#

6 Tests were analyzed here, a 10%,50% and 100% load test for a early adaoption scenario (for further info refer to the documentation)

In [3]:

```python
documents_10 = pd.read_csv(folder_path_logs + 'BillonDLT-RETEST-onChain-Docs-10.csv', delimiter=' ')
documents_100 = pd.read_csv(folder_path_logs + 'BillonDLT-RETEST-onChain-Docs-100.csv', delimiter=' ')
```

inspecting the logs to figure out needed data cleaning steps

In [4]:

```python
documents_10.head()
```

Out[4]:

		doc_hash	md5	pub_task_id	pub_address	cif	Dur_time	status	pub_end_time	threads	start_brg_time	create_brg_time	published_brg_time	dur_brg_time
0	ZKUN7ubbeyU5xRNyX5W2b7R53AU9eVAjueq7vWjGMRYc67...	acfb76938445ce6a31fbd25e9176e7c8	C64nNh8DtjanLB7vz	C64nNh8DtjanLB7vz4Py2uhL	http://116.202.72.122:17014	no_cif	159.328	FINISHED_OK	1.634496e+09	37	1.634496e+09	1.634496e+09	1.634496e+09	2.954
1	ZRaMaVW1iGwifm2NCPVhFWQ41Lz5Y1AbUNgKkM31yhuV8K...	77c35069e55ee3c8a6f2d97fb74be1a1	C5ZsUergoXymQ95qm	C5ZsUergoXymQ95qmjMeYtVP	http://116.202.72.122:17014	no_cif	159.308	FINISHED_OK	1.634496e+09	37	1.634496e+09	1.634496e+09	1.634496e+09	2.198
2	ZSBWiE2jRrskbCzUx1hadhy1cyngQo8VaxjAFyyVjWgqTb...	1aa7a7a7ba9b6667eeab17d0734ca249	CL28me6m44o85zLs9	CL28me6m44o85zLs93R173qU	http://116.202.72.122:17014		159.333	FINISHED_OK	1.634496e+09	37	1.634496e+09	1.634496e+09	1.634496e+09	2.429
3	Z2qQ7hYxh9FrMNvYFYRAQ722ujjDNPNfWr8jJFRkeWeNV1	d5f8287558fdd05cb6b17d6fe333c438	CDXihR3zG1n7uDtD	CDXihR3zG1n7uDtDSosgzzPp	http://116.202.72.122:17014	no_cif	159.431	FINISHED_OK	1.634496e+09	37	1.634496e+09	1.634496e+09	1.634496e+09	2.017
	ZTorvMn2Ga2	0dab370	CAC5	CAC597cX	http://11	no	159.43	FINIS	1.63449	37	1.634496	1.634496e	1.63449	1.18

4	dKRs1BEhf8zS7eC9cYS8ULKuPLDqaRtK9un	8e6473d24cecaef252f3eff06	97cXthfRr6NA4	thfRr6NA4b7s6AkS	6.202.72.122:17014	_cif	9	HED_OK	6e+09		e+09	+09	6e+09	1

In [5]:
```
documents_100.head()
```

Out[5]:

		doc_hash	md5	pub_task_id	pub_address	cif	Dur_time	status	pub_end_time	threads	start_brg_time	create_brg_time	published_brg_time	dur_brg_time
0	Z5Gbc3fBavbuudyUdTf8Krh1nWFcczXsxeotPeBiSw2WyV...	51589c483232c269176283076b7a5a21	C64nNh8DtjanLB7vz	CUXRDaffLSrxGnNFh4T7tCPZ	http://116.202.72.120:17006	no_cif	9.047	FINISHED_OK	1.634495e+09	10	1.634495e+09	1.634495e+09	1.634495e+09	1.149
1	ZDmkv7vrsSzD4iYxcPJ8raY5Z3Gj7QGb7yGK15R2ReDosg...	2e018c38c0be76f2d04a897c1151ba8c	C5ZsUergoXymQ95qm	CE45HzNxioujWc84T7en8dTX	http://116.202.72.120:17006	no_cif	9.168	FINISHED_OK	1.634495e+09	10	1.634495e+09	1.634495e+09	1.634495e+09	1.263
2	Z6fa6EVhp9iMXe6v9TJSjYcDcNzLFghaiJ3ARwyr7qC4Ps...	cb403c3afa60608273b56c2a912910ed	CL28me6m44o85zLs9	CW7dN6DnY8cGf5n4AVH8wGsF	http://116.202.72.120:17006		9.273	FINISHED_OK	1.634495e+09	10	1.634495e+09	1.634495e+09	1.634495e+09	1.151
3	ZPTwuCC516f9JUCssQQgR1pEivrSB74YdS7e2JZ15mrKcf...	6a3d59ee36d5f017ee647c6e23af3e76	CDXihR3zG1n7uDtD	CX1E8JNWNueJnxU2mZb4v2zH	http://116.202.72.120:17006	no_cif	9.404	FINISHED_OK	1.634495e+09	10	1.634495e+09	1.634495e+09	1.634495e+09	1.160
4	Z9X1Njwa4QXsWUtQnki1WFn2bZ21xFzoRxYeZDf7CDwV2t...	5ad12ed43719fe01d212c5aa0566a8de	CAC597cXthfRr6NA4	CFcTTMkphxpKn2475UHJEZo2	http://116.202.72.120:17006	no_cif	9.670	FINISHED_OK	1.634495e+09	10	1.634495e+09	1.634495e+09	1.634495e+09	1.097

convert to a format i can do the calcualtions with and do some data quality ceck

In [6]:
```
documents_10_count = documents_10['doc_hash'].count()
documents_100_count = documents_100['doc_hash'].count()
```

In [7]:
```
documents_10['START_TIME'] = pd.to_datetime(documents_10['start_brg_time'],unit='s')
documents_10['END_TIME'] = pd.to_datetime(documents_10['pub_end_time'],unit='s')
documents_100['START_TIME'] = pd.to_datetime(documents_100['start_brg_time'],unit='s')
documents_100['END_TIME'] = pd.to_datetime(documents_100['pub_end_time'],unit='s')
```

In [8]:

```
documents_10_start = documents_10['START_TIME'].min()
documents_10_end = documents_10['END_TIME'].max()
documents_100_start = documents_100['START_TIME'].min()
documents_100_end = documents_100['END_TIME'].max()
```

In [9]:

```
print("###################################")
print("documents test with 10% capacity")
print("duration:         "  + str(documents_10_end-documents_10_start))
print("start timestamp: "  + str(documents_10_start))
print("end timestamp:    "  + str(documents_10_end))
print("Number Tx:        "  + str(documents_10_count))
print("Tx/s:             "  + str(documents_10_count/(documents_10_end - documents_10_start)
print("###################################")
print("documents test with 100% capacity")
print("duration:         "  + str(documents_100_end - documents_100_start))
print("start timestamp: "  + str(documents_100_start))
print("end timestamp:    "  + str(documents_100_end))
print("Number Tx:        "  + str(documents_100_count))

print("Tx/s:             "  + str(documents_100_count/(documents_100_end -
                            documents_100__start).total_seconds()))
```

```
###################################

documents test with 10% capacity
 duration:              0 days 00:14:40.875000064
 start timestamp:       2021-10-17 18:43:33.494999808
 end timestamp:         2021-10-17 18:58:14.369999872
 Number Tx:             19200
 Tx/s:                  21.796509152830993
 ###################################
 documents test with 100% capacity
 duration:              0 days 00:14:06.090999808
 start timestamp:       2021-10-17 18:19:48.308000
 end timestamp:         2021-10-17 18:33:54.398999808
 Number Tx:             162391
 Tx/s:                  191.93089182124723
```

loading the server HW monitoring data

In [10]:

```
server=["dellR6515-01","dellR6515-02","dellR6515-03","dellR6515-04","dellR6515-05","dellR6515-06","de
llR6515-07","dellR6515-08","dellR6515-09","dellR6515-10"]
```

In [11]:

```
source_dir = r'C:\temp\billion_data_sorted\retest_18_logs'
```

In [12]:

```
def get_pd_frames(server):
    my_list = []

     for root, dirnames, filenames in os.walk(source_dir):
         for f in filenames:

             if server + ' Power' in f:

                 my_list.append(pd.read_csv(os.path.join(root, f)))

         concatted_df = pd.concat(my_list)

     return concatted_df
```

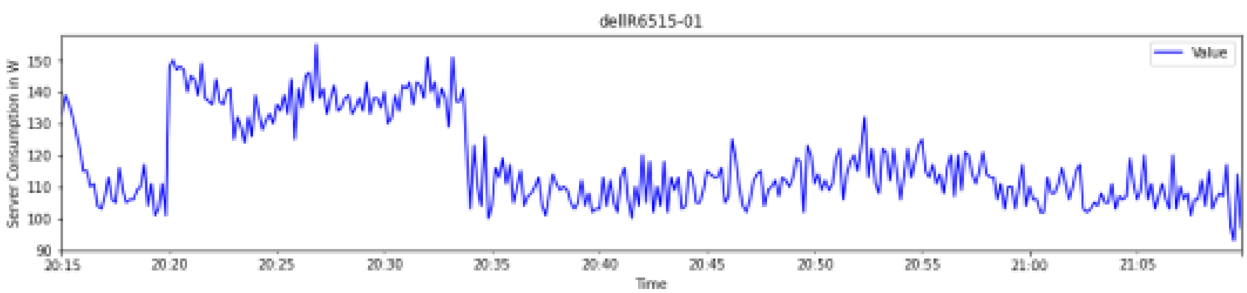
In [13]:

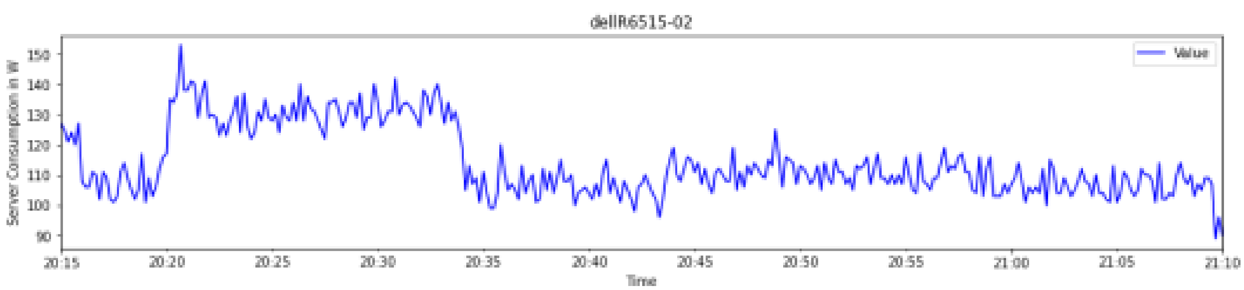
```
          d = {}

         for s in server:

             pd_temp = get_pd_frames(s)

             pd_temp[['Value','Var']] = pd_temp[': Instantaneous power reading'].str.split('
             ',expa pd_temp['Value'] = pd_temp['Value'].astype(int)
             '2021-10-11 21:30:00'

             pd_temp['Time'] = pd.to_datetime(pd_temp['Time'], format='%Y-%m-%d %H:%M:%S.%f')
             d[s]= pd_temp
```

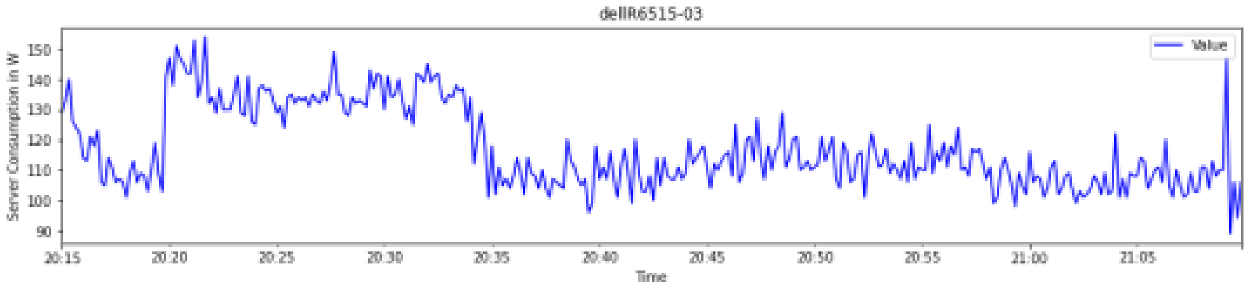
checking the data

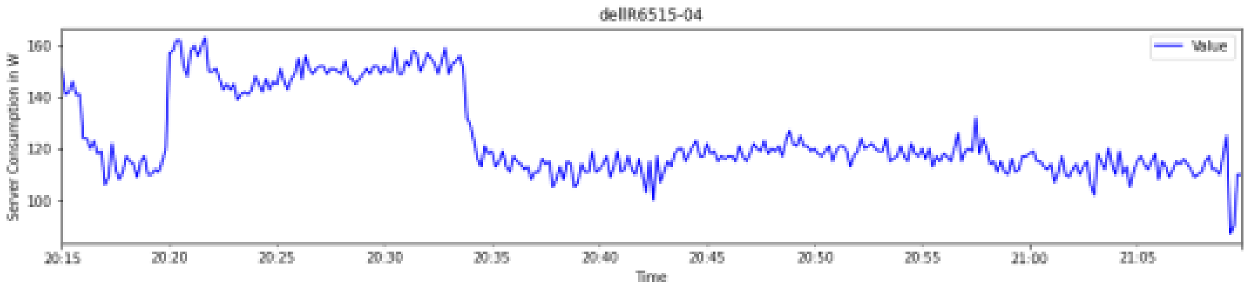
In [14]:

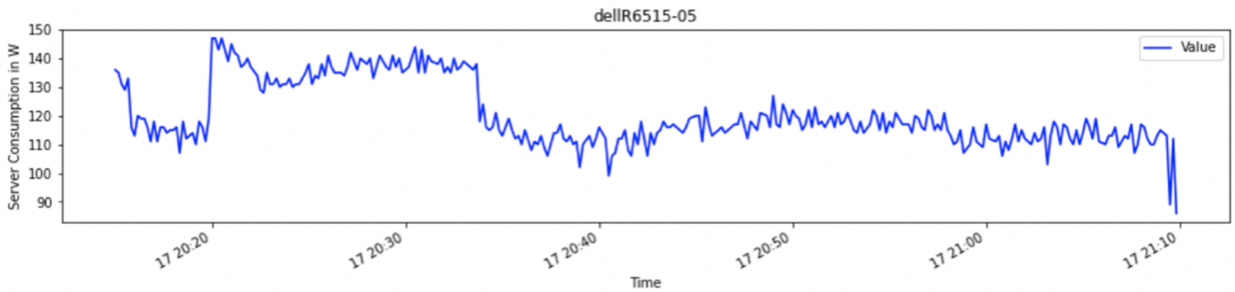
```
for s in server:

    ax = d[s].plot(kind='line',x='Time',y='Value',color='blue', title=s,figsize=(16,3))
    ax.set_ylabel("Server Consumption in W")
```

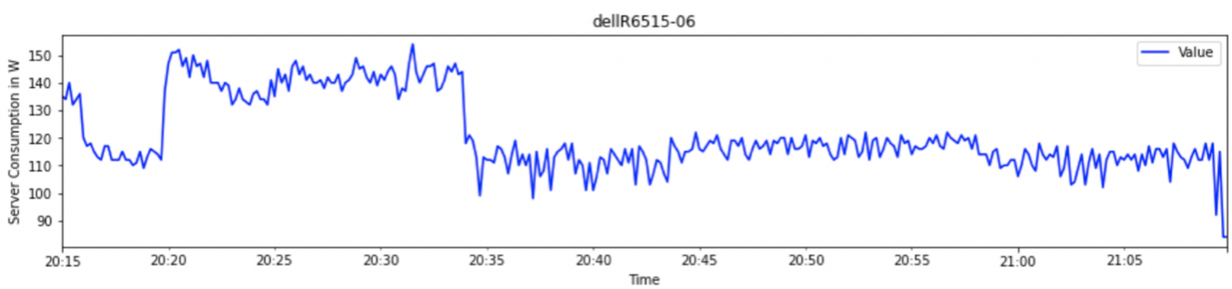


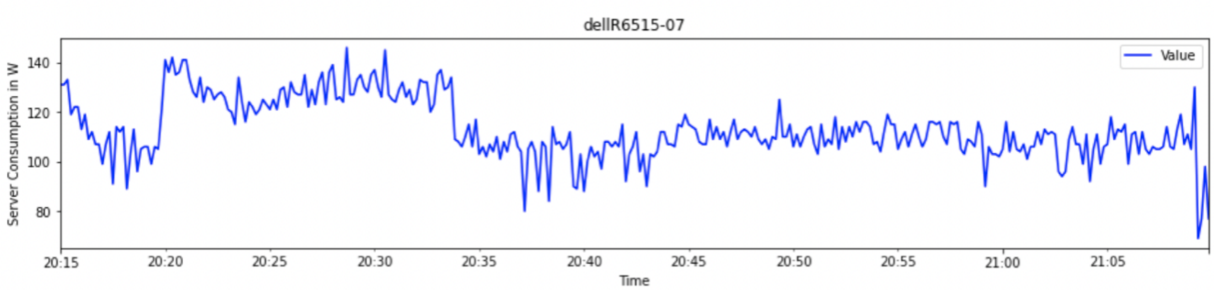


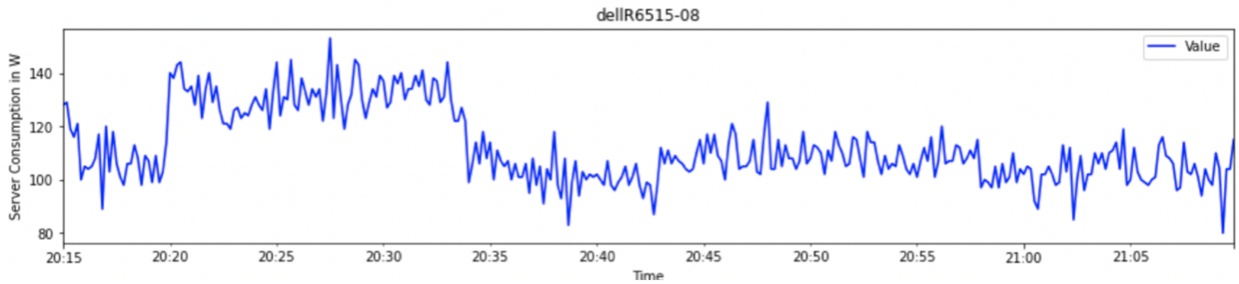

dellR6515-03
Server Consumption in W
Value
Time
20:15
20:20
20:25
20:30
20:35
20:40
20:45
20:50
20:55
21:00
21:05

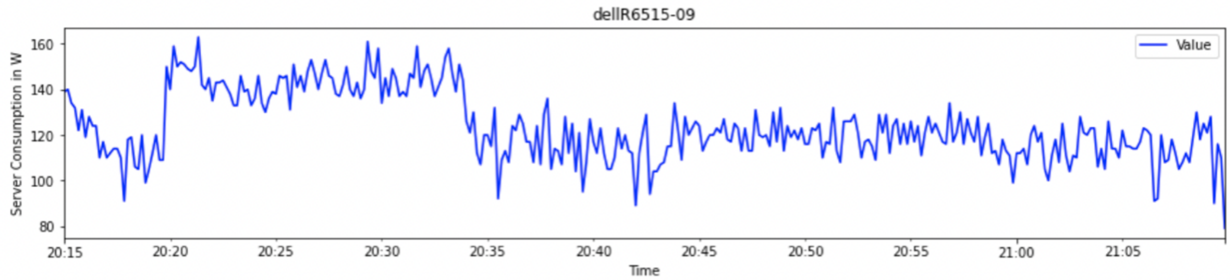

dellR6515-04
Server Consumption in W
Value
Time

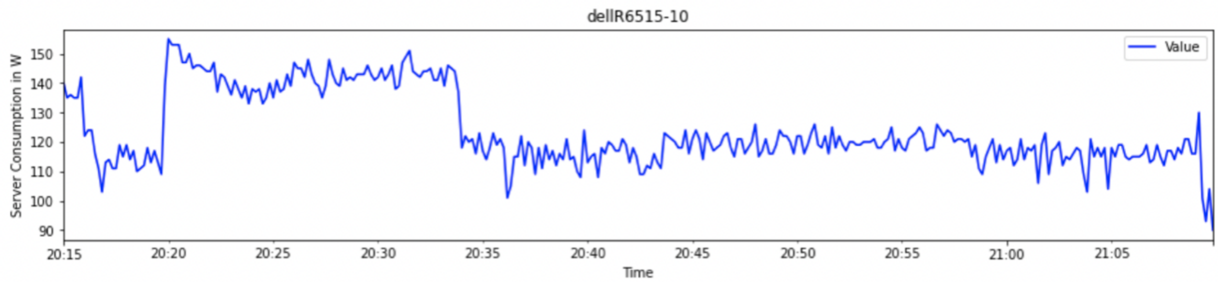

dellR6515-05
Server Consumption in W
Value
Time
17 20:20
17 20:30
17 20:40
17 20:50
17 21:00
17 21:10


dellR6515-06
Server Consumption in W
Value
Time


dellR6515-07
Server Consumption in W
Value
Time







In [15]:

```
results_billion = {}
```

In [16]:

```
documents_10_dict = {}

for s in server:

    df_temp = d[s][(d[s]['Time'] < documents_10_end + timedelta(hours=+2)) &
    (d[s]['Time'] documents_10_dict[s]= df_temp
```

In [17]:

```
used_energy_tx_Wh = {}

for s in server:

    mean_Wh =
    documents_10_dict[s]['Value'].mean()*(documents_10_end-documents_10_start).t
    used_energy_tx_Wh[s] = mean_Wh
```

In [18]:
```
print(used_energy_tx_Wh)
```

```
{'dellR6515-01': 0.0014495011763139205, 'dellR6515-02': 0.001414889248934659, 'dellR6515-0
3': 0.0014468944202769886, 'dellR6515-04': 0.0015177112926136364, 'dellR6515-05': 0.001497
829861111111, 'dellR6515-06': 0.00149425048828125, 'dellR6515-07': 0.001416192626953125,
'dellR6515-08': 0.001387807950106534, 'dellR6515-09': 0.0015382757013494318, 'dellR6515-1
0': 0.0015313243519176137}
```

In [19]:
```
results_billion["bil_d_10"] = (sum(used_energy_tx_Wh.values())/
len(used_energy_tx_Wh.valu results_billion["bil_d_10"]
```

Out[19]:
```
0.0014694677117858272
```

In [20]:
```
documents_100_dict = {}

for s in server:

    df_temp = d[s][(d[s]['Time'] < documents_100_end + timedelta(hours=+2))
    & (d[s]['Time'] > documents_100_start+ timedelta(hours=+2))]
        documents_100_dict[s]= df_temp
```

In [21]:
```
used_energy_tx_Wh = {}

for s in server:

    mean_Wh =
    documents_100_dict[s]['Value'].mean()*(documents_100_end-documents_100_start).total_
 seconds()/3600/documents_100_count
    used_energy_tx_Wh[s] = mean_Wh
```

In [22]:
```
print(used_energy_tx_Wh)
```

```
{'dellR6515-01': 0.00019899251285242196, 'dellR6515-02': 0.0001900023488423185,
'dellR6515-03': 0.00019655767676635222, 'dellR6515-04': 0.00021707500182728906,
'dellR6515-05': 0.00019760941069702138, 'dellR6515-06': 0.00020519027743514474,
'dellR6515-07': 0.00018613725939100513, 'dellR6515-08': 0.00019044504631251298,
'dellR6515-09': 0.00020840834750694931, 'dellR6515-10': 0.00020624594063330082}
```

In [23]:
```
results_billion["bil_d_100"] = (sum(used_energy_tx_Wh.values())/
len(used_energy_tx_Wh.values()))
results_billion["bil_d_100"]
```

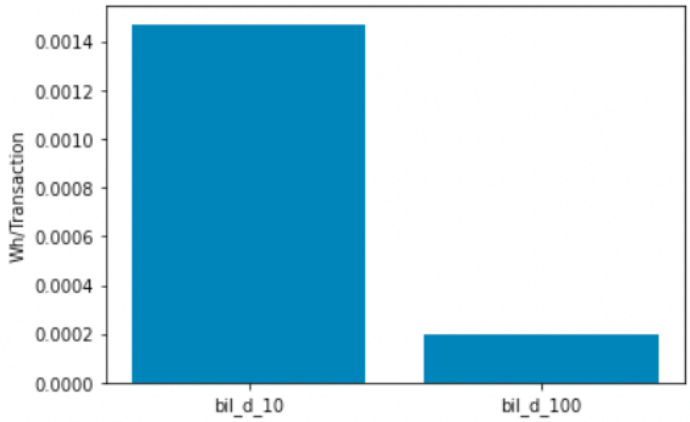
Out [23]:
```
0.00019966638222643096
```

In [24]:
```
import matplotlib.pyplot as plt
```

In [25]:
```
keys = results_billion.keys()
```

```
values = results_billion.values()
plt.ylabel("Wh/Transaction")
plt.bar(keys, values)
```

Out [25]: <BarContainer object of 2 artists>



In [26]:

```
results_billion
```

Out[26]: {'bil_d_10': 0.0014694677117858272, 'bil_d_100': 0.00019966638222643096}