Estimating transaction cost for cloud-based private ethereum blockchains

Igor Gonçalves Silva[1], **Pedro Henrique Gonzalez**[1], **Diogo Silveira Mendonça**[1]

[1]Federal Center for Technological Education of Rio de Janeiro (CEFET/RJ)
Rio de Janeiro – RJ – Brazil

igor.goncalves@aluno.cefet-rj.br,{pedro.silva,diogo.mendonca}@cefet-rj.br

Abstract. *Blockchain technology is increasingly being used by several companies in the most varied sectors of the economy. The possibility of having decentralized applications (DApps) allows for the emergence of technological innovations such as cryptocurrencies and decentralized asset tracking applications. Many of these DApps are deployed in the cloud with Infrastructure as a Service (IaaS) payment model, in which the payment is made according to the use of the service. However, it is not simple to estimate the cloud infrastructure costs that a DApp will consume. Furthermore, correctly estimating infrastructure costs is essential to analyze the viability and develop business models for enterprise DApps. This work presents an experience report on estimating the cloud infrastructure cost for an enterprise DApp. To do that, we deployed a private Ethereum DApp, using Proof-of-Authority consensus algorithm, with several different configurations of Amazon Web Services (AWS) EC2 instances and blockchain parameters. We benchmark the transaction processing capacity, CPU and disk usage in each configuration, estimating their maximum capacity and costs. We shared our methodology to measure and estimate those costs and our insights on best configuration practices for reducing costs of deploying enterprise DApps in the cloud.*

Resumo. *A tecnologia Blockchain é cada vez mais utilizada por diversas empresas dos mais diversos setores da economia. A possibilidade de ter aplicativos descentralizados (DApps) permite o surgimento de inovações tecnológicas como criptomoedas e aplicativos de rastreamento de ativos descentralizados. Muitos desses DApps são implantados na nuvem com modelo de pagamento Infraestrutura como Serviço (IaaS), em que o pagamento é feito de acordo com a utilização do serviço. No entanto, não é simples estimar os custos de infraestrutura em nuvem que um DApp consumirá. Além disso, estimar corretamente os custos de infraestrutura é essencial para analisar a viabilidade e desenvolver modelos de negócios para DApps corporativos. Este trabalho apresenta um relato de experiência na estimativa do custo de infraestrutura em nuvem para um DApp empresarial. Para isso, implantamos um Ethereum DApp privado, usando o algoritmo de consenso de Prova de Autoridade, com várias configurações diferentes de instâncias EC2 e parâmetros de blockchain da Amazon Web Services (AWS). Nós avaliamos a capacidade de processamento de transações, uso de CPU e disco em cada configuração, estimando sua capacidade máxima e custos. Compartilhamos nossa metodologia para medir e estimar esses custos e nossas intuições sobre as melhores práticas de configuração para reduzir os custos de implantação de DApps corporativos na nuvem.*

1. Introduction

Blockchain technology was initially developed to enable cryptocurrencies [Nakamoto et al. 2008]. The growth of cryptocurrencies made companies interested in using blockchain for many other purposes, such as tracking digital assets [Crosby et al. 2016]. Applications that use blockchain infrastructure to provide trust in a decentralized manner are usually called decentralized applications (DApps). In DApps, information/transactions can be validated without a centralized authority [Buterin et al. 2013], implying the possibility of cost reduction. Several companies from the most variated sectors of the economy are currently studying the implementation of enterprise DApps [Crosby et al. 2016], i.e., DApps which are provided by a set of companies.

Enterprise DApps are usually hosted on the cloud, using infrastructures as a service (IaaS) payment model, also called pay-as-you-go [Ibm 2019, Oracle 2019, Microsoft 2019, Amazon 2019]. In this format, the fees are proportional to the disponibility of the server, its processing capacity, RAM available, and storage space used [Ibm 2019, Oracle 2019, Microsoft 2019, Amazon 2019]. Properly estimate infrastructure costs is essential to analyze the viability and to develop business models for enterprise DApps. Although there are some reports on the cost of smart contracts execution in DApps [Zhu et al. 2019, Al Omar et al. 2019, Schäffer et al. 2019], they neither provide complete information needed for estimating cloud infrastructure costs nor a methodology to perform those estimates for enterprise DApps.

In this work, we present an experience report on estimating cloud infrastructure costs for an enterprise DApp. We deployed an Ethereum DApp, using Proof-of-Authority consensus algorithm, with several different configurations of Amazon Web Services (AWS) EC2 instances and blockchain parameters. We modified the benchmark tool chainhammer [chainhammer 2020] to measure not only transaction processing capacity, but also CPU and disk usage in each configuration. In this way, we estimated the maximum processing capacity and computational resource utilization of each configuration. We developed and used a theoretical model to estimate the long-term cloud infrastructure costs given the maximum transaction workload on the blockchain network.

We shared our methodology to measure and estimate infrastructure costs of deploying enterprise DApps on the cloud. We also provide our insights on best configuration practices for reducing those costs. As future work, we suggest some improvements in blockchain technology that could reduce long-term infrastructure costs of enterprise DApps.

The remainder of this paper is structured as follows. Section 2 explains the background needed for understanding the experience report. Section 3 presents related work. Section 4 explains our methodology to select blockchain configurations, measure and estimate costs. Section 5 presents the results found. We discuss the results in Section 6. Finally, we conclude and present future works in Section 7.

2. Background

This section presents the background needed for understanding our measurement and estimation methodology used during the experience report. We present concepts about

blockchain technology[Nakamoto et al. 2008], Ethereum [Wood et al. 2014] and cloud computing.

2.1. Ethereum Blockchain

Blockchain is a technology for enabling trust in a decentralized system of transacting peer participants. In Blockchain, a distributed record storage technology (Distributed ledger) is shared and used. A set of validation participants verifies each transaction sent to the DApp. The transactions that reach the consensus of those validation participants, using a consensus algorithm, are registered on the blockchain network. The validation nodes on the network must contain the full transaction history of all validated transactions.

A set of transactions is stored on the network in a block, and each block also has the hash address of the previous block. Such a process forms a block list with a valid chronological order of all transactions on the network. The transactions presented in the block are usually stored in a dispersion tree, as it easily allows checking if a transaction is present in the block [Merkle 1980].

The main feature of the Ethereum platform is the ability for users to program self-executing protocols that are used to facilitate or reinforce a contract between two or more parties in the Ethereum network. This protocol is called a smart contract, and its code is executed by Ethereum Virtual Machine [Wood et al. 2014].

Smart contracts can be implemented using several programming languages. One of the most used on the Ethereum platform is Solidity [Solidity 2019] language. An example of a smart contract would be selling a title deed from one party to another. One party would pay several installments of the sale price to the other, and the contract (which transfers the title to the other party) will be executed automatically when all installments are paid [Wood et al. 2014].

A consensus protocol is used to define which block will be added to the blockchain network and who will add it. The most common consensus algorithm in public blockchain networks, i.e., where anyone can send transactions and read data, is the proof-of-work. The Ethereum platform uses a modified version of the proof-of-work protocol called Ethash. This algorithm includes selecting random pieces of the dataset and using them in a hash function until the resulting value is equal to the number of zeros defined by the difficulty of the block [Wood et al. 2014].

Proof-of-work algorithms are not appropriate for enterprise blockchains since the peers are known and usually there is no competition or rewards for blocks validation. One standard consensus protocol in enterprise blockchains is proof-of-authority. In this algorithm, there is no computational cost for calculating hashes for mining (as in the proof-of-work). The transactions are collected, validated, and inserted by specific sealing nodes [Gethclique 2017].

Click is one of the implementations of this algorithm. In this implementation, it is possible to set a time period in which a new block can be sealed, limiting the number of transactions processed. This algorithm has the concept called sealing in order, which implies that the blocks must be sealed by a list of sealing nodes in which their addresses are in lexicographic order. For example, the sealer node in the first position of this list must seal the first block of the network and so on, the second sealer must seal the second

block, and so on, with the list of sealers being repeated when reaching the end of the list [Gethclique 2017].

If a block is sealed by a node that respects this concept, that block is considered an ordered block and otherwise a disorderly block. The network with the largest number of ordered blocks will be selected and chosen by the consensus algorithm. Another factor that will influence the selection of blocks is that a sealer can only seal one block on each *floor(number of sealers / 2) + 1* blocks. This means that at least 51% of the sealing nodes must be active in the network and sealing so that the network does not block [Gethclique 2017].

These sealing nodes can be defined in the genesis block and can be included/excluded by other sealers in a proposal that can be made for each sealed block. If this proposal is accepted by 51% of the sealing nodes, it is carried out. Proof-of-authority algorithms are widely used in private networks, in which the block inclusion process must be done quickly [Gethclique 2017].

2.2. Cloud Computing

Scalability is one of the main problems of companies that work with computing. A computing infrastructure can quickly become obsolete as a company grows and its computational demands increase. To solve this problem, cloud computing emerged, which is offering computing infrastructure as a service [JoSEP et al. 2010].

Cloud computing consists of creating a virtual machine that simulates specific hardware. Cloud users can use that machine in order to outsource computing/data storage. This outsourcing occurs when sending a request on the network, with the information that will be computed/stored in the cloud, so that the infrastructure in the cloud meets this request [JoSEP et al. 2010].

Usually, cloud infrastructure providers work with a payment system where the user pays as she consumes (pay-as-you-go). This form of payment is very interesting for users with regard to scalability, considering that to increase the computing/storage load, the user would only have to pay for the infrastructure made available [JoSEP et al. 2010].

3. Methodology

This section presents the smart contract under evaluation and the methodology used to measure and calculate the costs involved in using it in a private ethereum network deployed in the cloud.

3.1. Smart Contract under evaluation

In order to perform cost evaluations, we develop a smart contract to store/retrieve financial transactions, allowing users to record their financial transactions on the blockchain and later consult them. This contract was implemented using the Solidity language[Solidity 2019].

The evaluation used the *insertTransaction* method that stores the user's transaction and associates it with an id, which identifies a transaction. This id is returned as a method response. The smart contract code is available in Github [1].

[1] github.com/igorgs-rj/chainhammer/blob/master/hammer/contract.sol

Tabela 1. Input variables

Input Variable
1. Block period
2. AWS node type
3. Number of AWS nodes
4. Number of blockchain node instances in each AWS node
5. Number of threads

3.2. Tooling

In order to measure some parameters of blockchain infrastructure usage, we modified the benchmark tool called Chainhammer [chainhammer 2020], which aims to study the performance of a smart contract in the cloud.

Chainhammer measures the number of transactions per second (TPS) that an ethereum private blockchain network can process by sending many transactions at once and measuring the processing results. The modification made in this software was to allow the measurement of average CPU usage and disk consumption during the experiments. These data is necessary for the cost study of a smart contract.

3.3. Context and Variables

There are three types of blockchains regarding access control: private, permissioned and public. In this work, we chose to use a private network as it allows total control of the experiments and has no processing involved in authentication and authorization. Hence, we measured the cost of the application only (smart contract execution).

Among the various infrastructure cloud providers, we selected Amazon Web Services (AWS) to deploy the blockchain. We selected AWS because they have the cheapest pricing, are the most commonly used provider, and have good documentation. We used different types of AWS EC2 instances to estimate the cost of a blockchain application in the cloud. The instances used in the experiments were of the standard type. We selected this type of instance instead of instances with GPU processors with more processing capacity due to the project's financial limitations.

In order to evaluate the transaction costs, we considered a set of input and output variables. Table 1 presents the input variables. As explained in Section 2, the Clique algorithm accepts as parameter the period of time that each block will be sealed, which is called block period. This period could influence the throughput of transactions sealed per second and consequently in the transaction cost. We also plan to evaluate whether the types, number of AWS nodes, number of blockchain nodes in each AWS instance, and the number of threads influence the transaction throughput.

The output variables are presented in table 2. We plan to measure the throughput of transactions (in transactions per second) and the usage of computational resources such as CPU and disk usage. Finally, we calculate the transaction cost with the formulas of section 3.4.

Tabela 2. Output variables

Output Variable
1. Throughput (Transactions per Second)
2. Average CPU consumption
3. Disk consumption

3.4. Costs Calculation

The costs of cloud computing consist of two components, the disk storage cost used by instances and the cost of the availability of the instances. Cloud providers offer different prices depending on the region where the infrastructure is located. Only the AWS Oregon region was used for this work, as their prices are lower than other regions.

The availability cost of an instance is constant and depends on the number and type of nodes. Table 3 presents the availability costs of AWS EC2 instance used in our work by the time when this paper was written. On the other hand, the storage cost increases linearly with the number of transactions stored, considering that transactions in blockchain will not be deleted. This implies a constant growth of storage costs. While writing this paper, AWS charges USD 0.1 per Gigabyte of data stored in an Oregon data center.

The Chainhammer tool sends many transactions and calculates the mean and peak throughput for processing transactions in an Ethereum blockchain. With the mean throughput (TPS_{mean}), we calculate the maximum number of transactions that a blockchain configuration can process in a month (TXM_{max}) by multiplying the mean throughput by the number of seconds in a month ($SEC_{month} = 2,592,000$), as shown in Formula 1.

$$TPM_{max} = TPS_{mean} * SEC_{month} \tag{1}$$

As cloud computing processing is charged by availability, the best usage of resources is when we use the full processing power of the nodes. In case of blockchain, to maximize resource usage, we need to process the maximum number of transactions given the cloud nodes available. In this scenario, we calculate the minimum cost of processing a transaction ($CTXP_{min}$) by dividing the availability cost of a node per month (AV_{cost}) by the maximum number of transactions that the blockchain network can process in a month (TPM_{max}), as showed in Formula 2. It worth noticing that this cost is for each cloud node in the blockchain, as each node has its availability cost.

$$CTXP_{min} = \frac{AV_{cost}}{TPM_{max}} \tag{2}$$

In a blockchain, the disk is effectively used when a block is validated and stored in the blockchain. Furthermore, each block can have a different number of transactions. In our case, all transactions in a block are of the same type. Hence, we can calculate the disk usage of a transaction (D_{tx}) by dividing the disk usage of an experiment (D_{exp}) by the number of transactions sent in the experiment (N_{tx}), as shown in Formula 3.

$$D_{tx} = \frac{D_{exp}}{N_{tx}} \tag{3}$$

The disk cost in cloud computing is charged per Gigabyte per month (D_{cost}). Furthermore, the data stored in the blockchain cannot be deleted. Thus, for calculating the total disk cost of a transaction (CD_{total}), we need to measure the disk usage of a transaction (D_{tx}) and multiply it by the expected number of months that the transaction will be retained in the blockchain ($NM_{lifetime}$), as shown in Formula 4. As an information system lifetime is around ten years to 20 years, we can assume an expected lifetime, in months, of around 180 months. It worth noticing that disk cost, as calculated here, is not considering the interest rate of the economy or inflation. If those factors should be considered for analysis, the disk cost calculation should use the present value of each monthly expenditure.

$$CD_{total} = D_{tx} * NM_{lifetime} * D_{cost} \tag{4}$$

Finally, the minimum total cost of a transaction ($CTX_{min-total}$), per node, in a blockchain is the minimum cost of processing ($CTXP_{min}$) summed with disk total cost (CD_{total}), as shown in Formula 5. It worth noticing that is possible to calculate other costs than the minimum one with the same formula set presented in this section. To perform this calculation, set the throughput parameter in formula 1 for the desired one and apply the result found in the other formulas.

$$CTX_{min-total} = CTXP_{min} + CD_{total} \tag{5}$$

3.5. Experiments

In order to investigate the resource utilization and calculate the transaction cost of the private Ethereum network, we planned and executed several experiments making variations of the input parameters shown in Table 1.

We individually investigated the impact of the block period, the number of threads and the number of nodes on the throughput, CPU usage and disk usage, performing the experiments configurations of tables 4, 5 and 6, respectively. After performing those experiments, we selected the configuration that presented the best throughput for each type of cloud node instance and performed experiments varying the number of cloud node instances, as shown in Table 7.

4. Results

This section presents the results obtained in the experiments and calculates the transaction cost for the best and worst scenarios of them, showing its variation.

Table 8 presents the results of throughput by type of cloud node and block period. We can observe that for all types of cloud nodes, the period of 1 second had the best throughput. For this reason, the following experiments had the block period set to 1 second.

Table 9 presents the result of the experiments varying the number of threads in a blockchain node. We observed few variations in the throughput by increasing the number of threads used by a validation node. For all types of cloud instances, the best throughput was using one thread. For this reason, we set the number of threads with the value of one for the subsequent experiments.

Tabela 3. Availability cost of AWS EC2 instances

Type	Monthly cost (USD)
medium	33.408
large	66.816
xlarge	133.632

Tabela 4. Experiments varying the block period

Block Period
1
2
3
4
5
10

Tabela 5. Experiment varying the number of threads

Threads	Instance	Experiment code
1	Medium	1t-1m
2	Medium	2t-1m
3	Medium	3t-1m
4	Medium	4t-1m
1	Large	1t-1l
2	Large	2t-1l
3	Large	3t-1l
4	Large	4t-1l
1	Xlarge	1t-1xl
2	Xlarge	2t-1xl
3	Xlarge	3t-1xl
4	Xlarge	4t-1xl
5	Xlarge	5t-1xl
6	Xlarge	6t-1xl

Tabela 6. Experiment varying the number of nodes

Nodes	Instance	Experiment code
1	Medium	1t-1m
2	Medium	1t-2m
3	Medium	1t-3m
4	Medium	1t-4m
1	Large	1t-1l
2	Large	1t-2l
3	Large	1t-3l
4	Large	1t-4l
1	Xlarge	1t-1xl
2	Xlarge	1t-2xl
3	Xlarge	1t-3xl
4	Xlarge	1t-4xl
5	Xlarge	1t-5xl
6	Xlarge	1t-6xl

Tabela 7. Experiment varying the number of instances

Quantity	Instance	Experiment code
1	Medium	1m
2	Medium	2m
3	Medium	3m
1	Large	1l
2	Large	2l
3	Large	3l
1	Xlarge	1xl
2	Xlarge	2xl
3	Xlarge	3xl

Table 10 presents the results of the experiments varying the number of blockchain nodes. We can observe that increasing the number of blockchain nodes in the same EC2 instance increases the CPU utilization of the instance but can decrease the throughput. We observed this behavior in medium and large instances. In extra-large instances, the number of nodes increased until the value of 6 nodes did not decrease the throughput of the nodes.

Table 11 presents the results of the experiments varying the number of cloud node instances. We can observe that the throughput had few variations with the increase of the number of cloud node instances in the blockchain network. In the case of extra large instances, the throughput increased for two nodes and after decreased. In all other cases, the throughput decreased.

As explained in section 3.4, the transaction cost using cloud computing is influenced by the throughput of the blockchain network. In our experiments, the best throughput was around 230 TPS (xl-2i), and the worst was around 178 TPS (m-4n). The difference between these two throughputs is 48 TPS. The transaction cost per node of the best throughput result was USD 0.00000589, while the cost in the worst throughput scenario was USD 0.00000574. In the next section, we discuss the implication of the results found.

5. Discussion

In this section, we discuss the implications of the results found.

The throughput analysis showed that it is possible to use low-price cloud instances to deploy a private ethereum blockchain, with a considerable throughput, around 200 TPS, to store financial transactions. For example, if the worst throughput scenario (178 TPS) were maintained by one month, the blockchain would process around 462,000,000 transactions, while the best one (230 TPS) would process around 600,000,000 transactions. This capacity is enough to process almost all transactions of bitcoin history (650,000,000 by July of 2021) in a month.

The transaction cost found for the financial DApp is meager for the maximum throughput of the blockchain configurations analyzed, around USD 0.000006. However, it is highly improbable that nowadays, any private ethereum blockchain needs such high throughput. In this case, let us calculate the transaction cost for a more realistic scenario, such as 1.000 transactions/month. In this case, using an AWS EC2 medium instance, each transaction would cost USD 0.0335, which is an affordable price.

One interesting finding related to transaction cost is that the best throughput configuration is not always the best transaction cost. For example, the best throughput in our experiments costs USD 0.00000589 per transaction, while the worst one costs USD 0.00000574. This is because of the difference in the pricing of cloud servers used. This finding opens room for cost optimization in private blockchain networks instead of only performance optimization.

Another interesting finding concerning transaction cost is its composition in high throughput scenarios. In this case, the disk cost is much higher than the processing cost. For example, in our best throughput scenario, the processing cost is USD 0.00000006 while the disk cost is USD 0.00000567, i.e., the disk cost is two orders of magnitude higher than the processing cost. This shows the necessity for storage improvements in

Tabela 8. Throughput by type of cloud node and block period

Exp. code	TPS
m-1p	208,3600
m-2p	205,5146
m-3p	195,1863
m-4p	205,3500
m-5p	198,8600
m-10p	194,1300
l-1p	222,1948
l-2p	203,6118
l-3p	195,8390
l-4	200,9800
l-5p	193,4100
l-10p	195,1300
xl-1p	223.1560
xl-2p	210,8809
xl-3p	198,8580
xl-4p	202,0600
xl-5p	205,3100
xl-10p	191,1700

blockchain since it is impossible to delete or move data to cheaper storage like tapes.

Finally, if any DApp needs a throughput higher than 200 TPS, there is room for scaling by improving the processing capacity of the cloud node. It is also possible to make technological improvements, since none of our experiments achieved a complete utilization of the CPU capacity of the nodes.

6. Related work

Our objective was to measure and report the experience of calculating the transaction cost of a specific financial DApp. Many other studies benchmarked private ethereum networks, but they did not calculate the transaction cost of the DApp using cloud computing. Furthermore, the direct comparison of our results with other works is not applicable since the DApp measured and experiment configurations are different. The related work that is near to our experiments is explained below.

Schäffer et al. 2019 did a study on the impact that the variation of parameters (e.g., block frequency, block size) of a private blockchain network (using the Ethereum platform with the Geth client) would have on what concerns the performance and scalability of the network. Several experiments were carried out using instances of type c2 (large, xlarge, 2xlarge and 4xlarge) with varying number of nodes in order to measure the rate of successful transactions per second, the latency of the network and how the variation of these experiments would imply scalability of the network as a whole.

7. Conclusion

Blockchain technology is increasingly being used by several companies in the most varied sectors of the economy. In this work, we presented an experience report on estimating

Tabela 9. Experiments varying the number of threads and cloud node types

Exp. code	Bytes/TX	Cpu (%)	TPS
m-1t	338.0055	56.1	208.3600
m-2t	338.0046	56.4	204.6794
m-3t	338.0051	56.8	203.5053
m-4t	338.0048	56	207.0956
l-1t	338.0008	28	222.0920
l-2t	338.0034	56.5	214.4590
l-3t	338.0035	55.4	217.5400
l-4t	338.0020	55.4	217.4469
xl-1t	338.0008	28	223.1560
xl-2t	338.0037	27.6	219.8260
xl-3t	338.0032	27.4	222.7810
xl-4t	338.0014	27.6	220.4340
xl-5t	338.0018	27.6	203.6800
xl-6t	338.0005	27.6	204.2800

Tabela 10. Experiments varying the number of blockchain nodes

Exp. code	Bytes/TX	Cpu (%)	TPS
m-1n	338.0055	56.1	208.3600
m-2n	338.0051	63.8	195.6860
m-3n	337.9989	68.5	188.1014
m-4n	333.9943	74.1	178.3319
l-1n	338.0008	28	222.0920
l-2n	337.9997	62.3	204.4990
l-3n	337.9980	68.7	197.5973
l-4n	334.5656	72.4	187.8780
xl-1n	338.0008	28	223.1560
xl-2n	338.0029	31.6	217.5520
xl-3n	337.9966	35.2	209.5890
xl-4n	337.9944	38	211.8110
xl-5n	337.9907	43	223.8633
xl-6n	337.9927	44.3	223.2100

Tabela 11. Experiments varying the number of cloud node instances

Exp. code	Bytes/TX	Cpu (%)	TPS
m-1i	338.0055	56.1	208.3600
m-2i	327.0156	59.2	204.4173
m-3i	337.9964	58.9	205.9880
l-1i	338.0008	28	222.0920
l-2i	317.7832	51.1	218.1773
l-3i	340.4081	56.5	214.7089
xl-1i	338.0032	27.4	222.7810
xl-2i	337.9996	24.9	230.2542
xl-3i	337.9823	29.4	214.2000

cloud infrastructure costs for an enterprise DApp. We deployed an Ethereum DApp, using Proof-of-Authority consensus algorithm, with several different configurations of Amazon Web Services (AWS) EC2 instances and blockchain parameters. We measure the throughput of those configurations to investigate the best configuration scenario and calculated the transaction cost of the DApp. We shared our insights and methodology to estimate infrastructure costs of using enterprise DApps on the cloud.

Referências

[Al Omar et al. 2019] Al Omar, A., Bhuiyan, M. Z. A., Basu, A., Kiyomoto, S., and Rahman, M. S. (2019). Privacy-friendly platform for healthcare data in cloud based on blockchain environment. *Future Generation Computer Systems*, 95:511–521.

[Amazon 2019] Amazon (2019). Amazon blockchain platform pricing https://aws.amazon.com/pt/managed-blockchain/pricing/.

[Buterin et al. 2013] Buterin, V. et al. (2013). Ethereum white paper. *GitHub repository*, 1:22–23.

[chainhammer 2020] chainhammer (2020). Projeto chainhammer - https://github.com/drandreaskrueger/chainhammer.

[Crosby et al. 2016] Crosby, M., Pattanayak, P., Verma, S., Kalyanaraman, V., et al. (2016). Blockchain technology: Beyond bitcoin. *Applied Innovation*, 2(6-10):71.

[Gethclique 2017] Gethclique (2017). Clique poa protocol - https://github.com/ethereum/eips/issues/225.

[Ibm 2019] Ibm (2019). Ibm blockchain platform pricing https://www.ibm.com/cloud/blockchain-platform/pricing.

[JoSEP et al. 2010] JoSEP, A. D., KAtz, R., KonWinSKi, A., Gunho, L., PAttERSon, D., and RABKin, A. (2010). A view of cloud computing. *Communications of the ACM*, 53(4).

[Merkle 1980] Merkle, R. C. (1980). Protocols for public key cryptosystems. In *1980 IEEE Symposium on Security and Privacy*, pages 122–122. IEEE.

[Microsoft 2019] Microsoft (2019). Microsoft blockchain platform pricing https://azure.microsoft.com/en-in/pricing/details/blockchain-service/.

[Nakamoto et al. 2008] Nakamoto, S. et al. (2008). Bitcoin: A peer-to-peer electronic cash system.

[Oracle 2019] Oracle (2019). Oracle blockchain platform cloud service pricing https://www.oracle.com/br/blockchain/.

[Schäffer et al. 2019] Schäffer, M., Di Angelo, M., and Salzer, G. (2019). Performance and scalability of private ethereum blockchains. In *International Conference on Business Process Management*, pages 103–118. Springer.

[Solidity 2019] Solidity (2019). Solidity language https://solidity.readthedocs.io/en/v0.5.11/.

[Wood et al. 2014] Wood, G. et al. (2014). Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper*, 151(2014):1–32.

[Zhu et al. 2019] Zhu, L., Wu, Y., Gai, K., and Choo, K.-K. R. (2019). Controllable and trustworthy blockchain-based cloud data management. *Future Generation Computer Systems*, 91:527–535.