Performance Analysis of Decentralized Physical Infrastructure Networks and Centralized Clouds

[redacted]*, [redacted]*†, [redacted]†, [redacted]*
*[redacted]
[redacted]
[redacted]
†[redacted]

***Abstract*—The advent of Decentralized Physical Infrastructure Networks (DePIN) represents a shift in the digital infrastructure of today's Internet. While Centralized Service Providers (CSP) monopolize cloud computing, DePINs aim to enhance data sovereignty and confidentiality and increase resilience against a single point of failure. Due to the novelty of the emerging field of DePIN, this work focuses on the potential of DePINs to disrupt traditional centralized architectures by taking advantage of the Internet of Things (IoT) devices and crypto-economic design in combination with blockchains. This combination yields a more distributed, resilient, and user-centric physical infrastructure deployment. Through comparative analysis with centralized systems, particularly in serverless computing contexts, this work seeks to lay the first steps in scientifically evaluating DePINs and quantitatively comparing them in terms of efficiency and effectiveness in real-world applications. The findings suggest DePINs' potential to *(i)* reduce trust assumptions and physically decentralized infrastructure, *(ii)* increase efficiency and performance simultaneously while improving the computation's *(iii)* confidentiality and verifiability.**

I. Introduction

Historically, industries such as telecommunications were centralized and dominated by a few centralized authorities. The rise of the Internet fundamentally challenged this centralization of communication technologies, initially spurring a shift toward decentralization. However, this movement has seen a rebound towards centralization, especially evident in the domain of cloud computing, where monopolization was further accelerated *e.g.,* through economies of scale [1].

The monopolization of cloud providers has led to centralized trust in cloud providers that govern vast silos of confidential data [2]. Large-scale data breaches and leaks [3] highlighted these implicit trust assumptions and have shown that the future infrastructure needs to be built on less centralized dependencies.

The emergence of Decentralized Physical Infrastructure Networks (DePIN) signals a significant shift driven by the need to improve data sovereignty, reduce trust centralization, and increase resilience against single points of failure. This evolution reflects broader technological trends that demand increased confidentiality in computing [4].

At the core of this shift, DePINs propel a novel paradigm that merges the Internet of Things (IoT) with blockchain technology and crypto-economic design (*i.e.,* tokenomics) to design a more distributed and resilient approach to digital infrastructure. One example is to take advantage of underutilized resources, such as upcycling of phones [5].

The significance of DePINs extends beyond the technical domain, proposing a radical change in how users engage with physical infrastructures. From solar panel-equipped homes to decentralized versions of Google Maps built from user-contributed data (*e.g.,* Hivemapper [6]), DePINs encapsulate a vision of the future where technology enables a more inclusive and collaborative approach to building and maintaining the very backbone of society [7].

A key concern with DePINs are scalability and performance properties, mainly due to *(i)* the large number of smart devices involved, *(ii)* and the performance with respect to additional latency introduced by interactions with blockchains [8]. For that reason, empirical analysis and comparison of DePINs with centralized CSPs are crucial.

Despite the promising advantages of DePINs, the body of research dedicated to exploring these networks needs to be expanded, highlighting a compelling opportunity for scientific contribution. This work addresses the critical gaps identified in the background and related literature review, specifically the need for empirical validations and performance comparisons with centralized systems. This work focuses on deploying DePINs for computing resources, particularly within a serverless computing model, to then compare them to centralized CSPs.

This work focuses on laying the foundations for comparing DePINs quantitatively with centralized approaches. By providing a direct performance comparison with traditional centralized approaches (*e.g.,* Google Cloud Platform (GCP)), this work aims to substantiate the theoretical advantages of DePINs with empirical evidence. Exploring DePINs in compute-intensive scenarios is particularly crucial, as it promises to reveal new insights into their feasibility, efficiency, and effectiveness in real-world applications.

The remainder of this paper is organized as follows. Section II presents the relevant background and related work, emphasizing the novelty and challenges of DePINs. Section III outlines the proposed solutions and methodologies to explore the potential of DePINs in the context of serverless computing. Section IV provides an evaluation and discussion of the findings, critically assessing the performance of DePINs against centralized alternatives. Finally, Section V summarizes and draws preliminary considerations.

II. Background and Related Literature

At the core, DePIN networks share the conceptual property of providing a token-based incentive for the shared provisioning of resources. The resources targeted by such an economically incentivized scheme refer to a wide variety of types and deployment scenarios, including file storage, computing power, energy, communication connectivity, or dedicated services [14]. Due to the novelty of the paradigm, definitions, and subsuming of related literature, research that contributes to DePIN is still sparse. For example, a recent survey has identified a handful of solutions that fulfill the conceptual model of decentralized infrastructure networks [9]. Importantly, a taxonomy of the architectural elements found in DePIN approaches has been proposed, preparing the path for scientific analysis of such solutions. Regarding the work at hand, four key elements are identified for the discussion of DePIN solutions: *(i)* physical hardware needed to participate in the economy, *(ii)* governance actions defined on real-life procedures, *(iii)* middleware to act as a broker between physical and digital components, and *(iv)* smart contracts executing the BC-based backbone of those networks.

The related literature on DePIN networks can be summarized into two areas. First, a small number of literary works have explicitly focused on DePIN from a research perspective. For example, based on a keyword search, the aggregating search engine *Google Scholar* yields only five results to the query comprising "decentralized physical infrastructure networks" or "DePIN." Four elements were published in 2023 and one in 2024 (*cf.* TABLE I), stressing the novelty and lack of research in this area. The second area is industry solutions that advertise their proposition under the term, or that could be considered DePINs under the taxonomy of [9]. For example, *Filecoin* [15] could be viewed as a DePIN. However, it is not explicitly advertised as such.

From the list of publications, TABLE I highlights that several papers have not only discussed theoretical aspects of DePIN, such as the taxonomy provided by [9]. For example, while [10] proposed a roll-up-centric architecture for smart devices, [11] designed a generic protocol for device verification in DePIN. Where [12] focus specifically on the Global Navigation Satellite System, [13] present a novel Reinforcement Learning-based (RL) incentivization scheme.

Although these literary works concentrate either on specific problems within DePIN or dedicate themselves to a specific scenario, none present results from practical experiments. From the perspective of the study at hand, none focus on DePIN for computing resources, especially when considering a serverless service model. Thus, no comparisons are made with centralized approaches. All of these claims are substantiated by the fact that these publications are either white papers or published as non-peer-reviewed preprints. Although this is not a weakness per se, given the novelty of the paradigm, it calls for a stipulation of research in this field.

TABLE II: Related Industry Solutions

Solution	***Scope***	***Scientific Evaluation***
IoTeX	IoT, Smart Devices	None
Helium	Wireless Networks	None
IOTA	IoT Data	None
Streamr	P2P Infrastructure	None
MXC	Wireless Networks	None
Akash	Container Runtime	None
GEODNET	Satellite Navigation	None
Redacted	Serverless Computing	Comparative Performance Analysis

Regarding existing DePINs, notable networks include *IoTeX*, *Helium*, *IOTA*, *Streamr*, and *MXC* [11]. *IoTeX* and *IOTA* aim to interconnect IoT and smart devices, hence, not focusing on compute-heavy scenarios [16], [17]. In this aspect, *Helium* and *MXC* are similar since they focus on interconnecting wireless networks [18], [19]. Finally, while Streamr [20] focuses on infrastructure provisioning using a P2P model, *GEODNET* incentivizes the operation of a token-based localization system. Underpinning all of these approaches is the observation that none provides scientific evidence of their effectiveness and efficiency. This is magnified when focusing on the scenario of compute-intensive scenario in DePIN. Here, *akash* is the most closely related industry-driven effort. As described, users engage with Akash by procuring computing resources from cloud providers prior to deploying Docker containers on the platform. The marketplace records on-chain data about user requests, bids, lease agreements, and settlement payments. The blockchain infrastructure employed by Akash serves as an integral component. As reported, the platform is able to beat the cost of centralized providers. However, no scientific publications report on the performance of the platform.

Based on the above discussion leading to the summary in TABLE II, the following limitations are observed:

- ***l1:*** A lack of studies exploring DePIN solutions – focusing on compute resources, explicitly investigating the applicability of DePIN for serverless deployments.
- ***l2:*** A lack of studies directly comparing the performance of a DePIN with a centralized alternative.

TABLE I: Related Literature on DePINs

Solution	***Proposition***	***Resources***	***Evaluation***	***Comparison to CePIN***
[9], 2023	DePIN Taxonomy	N/A	Discussion	None
[10], 2023	Rollup-centric Architecture	Sensing, Smart Devices	None	None
[11], 2023	DePIN Device Verification Protocol	Generic	None	None
[12], 2023	Satellite Navigation Framework	Navigation	Market Analysis	Market Analysis
[13], 2024	RL-based Staking Incentivization	Generic	Simulation (IoTeX)	None
This, 2024	Mobile Serverless Computing Network	Compute	Comparative Performance Analysis	Cloud Service Providers

III. *Redacted* Decentralized Serverless Cloud

Widely recognized challenges of blockchains and the Internet, in general, are *(i)* the centralization of trust in auxiliary systems, *(ii)* the seamless and permissionless interoperability of fragmented ecosystems and *(iii)* the effectiveness and confidentiality of the execution layer. *Redacted* is a Layer-1 blockchain that addresses these shortcomings with a novel decentralized and serverless approach [5].

Redacted provides a modular separation of the consensus, execution, and application layer (*cf.* Fig. 1). The *Redacted* orchestrator is embedded in the consensus layer and contains a purpose-built reputation engine to ensure reliability and encourage honest behavior [5].

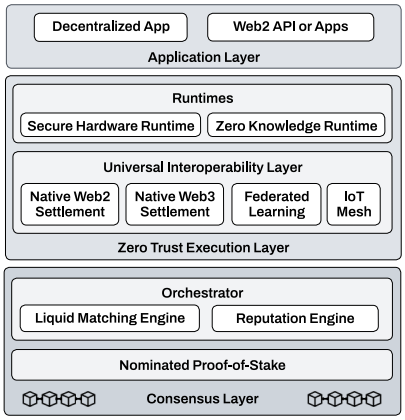

Fig. 1: *Redacted* Architecture [5]

So-called Consumers register their `job` on the orchestrator, offloading their computation to processors on the execution layer. Depending on the requirements, consumers can select suitable runtimes, *e.g.,* secure hardware runtimes based on smartphones' external coprocessors.

The *Redacted* execution layer takes advantage of secure hardware coprocessors, removing the trust required from third parties, and reducing them to cryptographic hardness assumptions and a single root of trust (*i.e.,* secure hardware with key attestation and a coprocessor). With this approach, *Redacted* takes advantage of smartphone hardware and achieves efficiency and confidentiality improvements compared to CSPs [21].

A. End-to-End Job Execution

The following description follows a `job` from definition and deployment to completion (*cf.* . Figure 2). First, during `Job` Registration (1) consumers define their `job` details, *e.g.,* at what destination the `job` should be *settled i.e.,* on which protocol the `job output` should be persisted (*e.g.,* Bitcoin Mainnet, Google Cloud).

During this step, the consumer must also define which processors the `job` should be executed, either *(a)* on personal processors, or *(b)* on selected, known processors (*e.g.,* known trusted entities), or *(c)* on public processors. For *(a)*, a processor reward is not required, as it is a permissioned setting. For *(b)* a reward is optional, and for *(c)* the liquid matching engine and the orchestrator will match processor resources with consumers' `jobs`.

In addition, more details of the `job` need to be declared, such as *scheduling* parameters, including start time, end time, the interval between executions, as well as the duration in milliseconds and the maximum start delay in milliseconds. Furthermore, specific resource management parameters, such as memory usage, network requests, and storage requirements of the `job` need to be declared. Finally, the reward for the execution of the job should be declared, as well as the minimum reputation (only applies to *(c)* public processors). Then the `job` will be persisted on the Consensus Layer and reaches `OPEN` state.

Second, the `Job` is acknowledged by the processor and fetches the details from the *Redacted* chain. Now the `job` reaches the `MATCHED` state, and no other processors will attempt to acknowledge it. Since `jobs` can have different scheduling configurations (*e.g.,* on demand, every minute, etc.). Therefore, if the processor acknowledges that all slots can be adhered to, the `job` reaches the `ASSIGNED` state.

Third, the `job` is executed on the processor runtime. In the illustrated example of Fig. 2, the execution is performed inside of the Secure Hardware Runtime, thus *confidentiality* is ascertained by secure hardware, such as an isolated and external coprocessor (*e.g.,* Google's Titan Chip [22]).

Once the `job` execution is completed, the output is delivered to the declared destination (`Job` Fulfillment (4)), which can be another Web2 system (*e.g.,* REST-API, FL model) or a Web3 system (*e.g.,* Tezos, Ethereum) that receives the output. In case of a cross-chain transaction, the processor settles the gas fees on the destination chain, since the consumer has locked the necessary reward and the gas fee amount up front when registering the `job`.

Finally, after completion, the processor reports back to the reputation engine. To ensure the reliability of the *Redacted* protocol, the reputation engine is continuously fed with reliability metrics, for example, right after the completion or failure of the job.

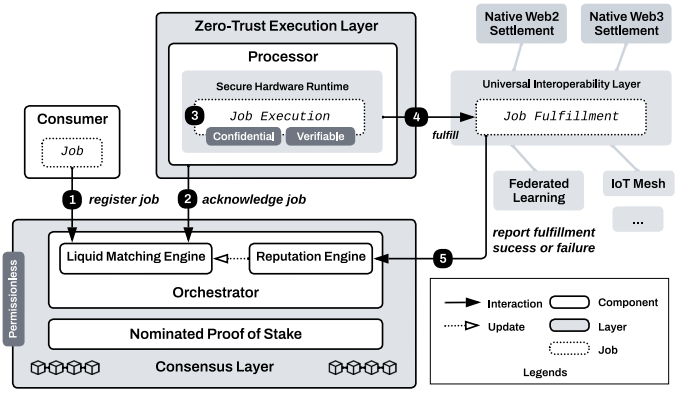

Fig. 2: End-to-End Zero Trust Job Execution [5]

IV. Experiments

Since DePINs are clearly a novel subject in scientific research [9] and decentralized networks are complex socioeconomic systems, a combination of empiricism [23] and experimental analysis is needed to investigate the applicability and effectiveness of the DePIN paradigm for serverless computing. Based on the systemization described in Section III, the evaluations aim to investigate the following research questions.

RQ1: Based on the description of the system architecture of *Redacted*, how many nodes have adopted the protocol and how are they distributed?

RQ2: Based on the identified set of nodes comprising the *Redacted* network, how does the provided end-user service compare with centralized cloud-based ones?

A. Network and Node Discovery

To understand the deployment and adoption of the DePIN answering ***RQ1***, an experiment is performed on node discovery. Due to the lack of a centralized management node in the network that would introduce a single point of failure and the absence of a governance function for node admission (*i.e.,* that allows anyone with the appropriate hardware to participate in the permissionless network), a different approach must be employed. Since a subset of the nodes participate in the serverless function provisioning service, this can be exploited to infer information about the nodes. To do so, a web-based service has been deployed in the Google Cloud Platform (GCP) [24]. While the RESTful endpoint does not execute any relevant load, it integrates with the logging functionality provided in the cloud service. Furthermore, a client in the *Redacted* service is provisioned, which leverages the compute network to access DePIN serverless functions. More in detail, the following steps are executed as part of the experiment:

1) Deploy RESTful endpoint in Google Cloud
2) Deploy client in the DePIN
3) Deploy a serverless request on DePIN to fetch a resource from the RESTful endpoint.
4) The executing node in the DePIN performs an HTTPS `GET` request on the endpoint.
5) The cloud service collects client information (*i.e.,* the DePIN executor) including source IPv4 or IPv6 address.
6) The client IP addresses discovered by the cloud service are aggregated – the above procedure is repeated at least n times, and, until no additional nodes are discovered

Following said procedure, 121 function deployments have been made, and based on a service success rate of 100%, the same number of logs were obtained in 187 seconds. After extracting the IPv4 and IPv6 addresses (*i.e.,* the DePIN nodes sending a request while executing the serverless function) from the logs, the subnets in which these addresses are announced in BGP were looked up using the BGP Looking Glass provided by [**?**], thus heuristically reducing the number of nodes having multiple IPv4 or IPv6 addresses from the same uplink (*i.e.,* accounting for address changes and multihoming). Then, the *whois* directory service was used to obtain the geographical locations from where the prefixes are announced [**?**]. While this still has a certain ambiguity, each subnet was analyzed individually. For example, if an IP address belongs to a network service provider (*e.g.,* VPN providers, cloud providers), it would have been eliminated. Ultimately, the distribution of countries depicted in Fig. 3 presents an overview of current geographical penetration.

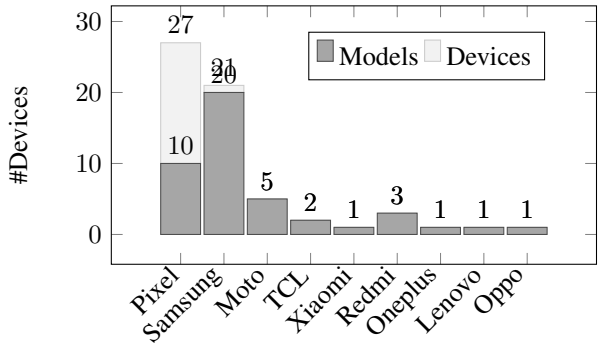

Fig. 4: Devices Employed in the *Redacted* DePIN

To answer ***RQ1***, 30 nodes have been discovered in 14 countries, effectively covering three continents (*i.e.,* Europe, Asia and North America). Although this analysis is not exhaustive (*i.e.,* presents only a snapshot of the deployed nodes), a longitudinal study could offer additional insight while requiring one to account for additional pitfalls (*e.g.,* address changes, multihoming). Furthermore, all function deployments have been successful, indicating that there were no Sybil

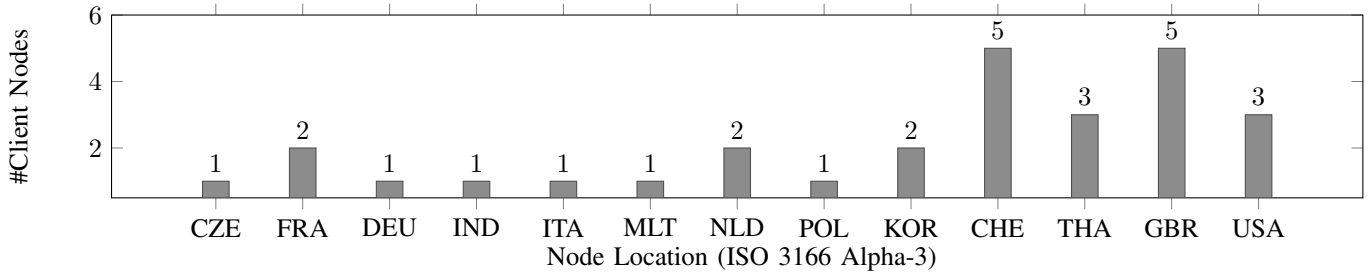

Fig. 3: Node Discovery

nodes at the time of the experiment. However, due to the limited complexity of the workload required to be performed by the serverless function, additional experiments are needed to investigate the behavior of the nodes.

Additional information on ***RQ1*** involves analysis of the nodes themselves. Here, the experiment relies on the `User-Agent` header set by the clients performing the requests on the Cloud instance. Although even client IP addresses do not inherently correctly identify the geographic location (*e.g.,* they could be tunneled), relying on the user agent information inherently implies that the client information is trusted. Nevertheless, with this trust assumption in mind, the following conversations are made. First, all nodes report to run the Dalvik virtual machine, as expected based on the system description in Section III. Second, 62 distinct devices are discovered. Relating this to the 30 unique IP addresses discovered implies that nodes employ ≈2 devices per node. Ten nodes operated *Redacted* on top of Android 11 (*i.e.,* 16%), 13 operated Android 12 (*i.e.,* 20%), 27 operated Android 13 (*i.e.,* 44%), and 12 devices used Android 14 (*i.e.,* 19%). As indicated in Fig. 4, the 62 devices comprise 43 different models across nine smartphone vendors.

B. Comparative Performance Analysis

In the second experiment, a computationally expensive workload is deployed to the following instances on the respective platforms, which enable serverless computation. In TABLE III, each platform is characterized in terms of deployment location and runtime environment. To each of these platforms, the workload depicted in Algorithm 1 is deployed. The algorithm employs the Sieve of Eratosthenes to efficiently identify and generate prime numbers up to a specified maximum value (*i.e.,* `max`). It initializes arrays to track whether the numbers are marked as composite (sieve) and to store prime numbers (primes). The algorithm iterates through each number, marking its multiples as a composite in the sieve array, and adding the unmarked numbers to the primes array. By systematically eliminating multiples, the algorithm efficiently identifies prime numbers within the given range, providing an optimized method for generating a list of primes up to the specified maximum value. The algorithm complexity is expressed as $O(n \log \log n)$.

TABLE III: Platforms Involved in Comparative Analysis

Platform	***Location***	***Runtime***	***Hardware***
Redacted	Distributed	V8 ECMA-262	Randomly Sampled
Local1	Local	Node.js v18	i7-8650U@1.90GHz, 16GB
Local2	Local	Node.js v18	7502P@2.5GHz, 128GB
Microsoft Azure	Iowa, USA	Node.js v18	Unknown
Amazon AWS	Sweden	Node.js v18	Unknown
Google Cloud	Iowa, USA	Node.js v18	Unknown

To execute the functions, a JavaScript adapter was written and deployed to each of the platforms. Then, a test runner, executes each function several times with `max=50,000,000`, measuring the completion time of the function. This value was selected since it was the largest one that was successfully completed across all platforms.

Algorithm 1 Benchmark Algorithm – Sieve of Eratosthenes

```
1:  function GETPRIMES(max)
2:      sieve ← []
3:      i, j ← 0
4:      primes ← []
5:      for i ← 2 to max do
6:          if not sieve[i] then
7:              PUSH(primes, i)
8:              for j ← i × 2 to max by i do
9:                  sieve[j] ← true
10:             end for
11:         end if
12:     end for
13:     return primes
14: end function
```

In Fig. 5, the distribution across all response times are plotted for each platform, respectively, based on the analysis of ≈1000 measurement samples. When comparing the average delay, it can be seen that the *Redacted* platform completes the previously described workload in 2790 ms, while AWS requires 3683 ms, and Google Cloud 5565 ms. Finally, Azure exhibits the highest delay with an average response time of 6102 ms. Thus, it can be observed that the globally decentralized network provided by the DePIN may indeed be a viable alternative for computationally intensive scenarios. Furthermore, it can be seen that the response times distribution through *Redacted* is narrowly distributed (*i.e.,* average of 2790 ms, with a standard deviation of ≈134), although it is supported by a heterogeneous and ungoverned physical infrastructure.

Answering ***RQ2*** by comparing the performance of *Redacted* with two centralized and local approaches (*i.e.,* not serverless), two interesting observations arise. As depicted

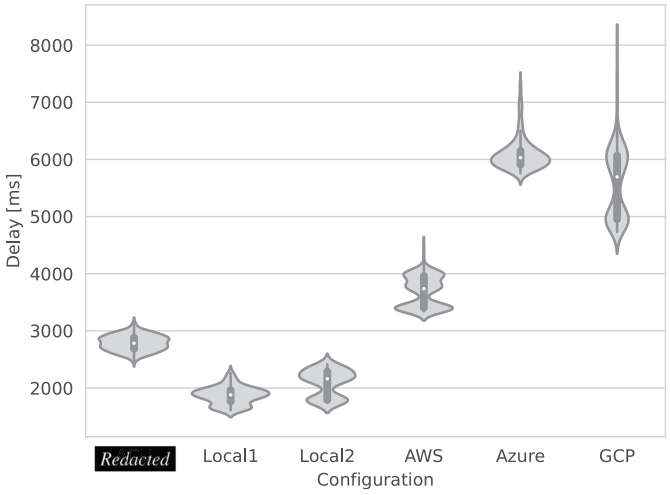

Fig. 5: CPU-intensive Benchmark [25]

in Fig. 5, both local configurations (*i.e., Local1* and *Local2*) outperform *Redacted* by a comparatively small margin. Compared to the mobile CPU on *Local1*, *Redacted* presents an increase of 30.55%, and an increase of 46.04% compared to the server. Nevertheless, when considering the ad hoc deployment model of *Redacted* (*i.e.,* without hardware or software maintenance required for a client), the solution presents a strong improvement compared to centralized solutions.

C. Power Efficiency

Aside from computational effectiveness, power efficiency is vital, hence, the performance in relation to power consumption must be considered. The previously described workload mainly benefits from single-threaded performance since I/O can be largely neglected. Using the server from *Local2* as an illustrative scenario, the workload completely saturates one CPU core for an average duration of 2092 ms. As described by the manufacturer, the CPU requires 180 Watts of power [26]. Ignoring the consumption of peripherals, each core would require 5.625 Watts. Thus, a single run of the benchmark algorithm would require 3.268750×10^{-3} Wh. In comparison, one Cortex-A715 core leveraged by *Redacted* requires less than 300 mW. When factoring in the extended computation time, one iteration would require 2.275833×10^{-4} Wh, marking a stark decrease in performance per watt.

In other words, with a single watt-hour, the workload above can be executed ≈4394 times using *Redacted* or ≈306 in a local scenario. Furthermore, it should be restated that this comparison draws on the values obtained in a local environment. Although cloud providers stress the importance of power efficiency [27], the authors have not identified any sources presenting actual numbers of the cloud provider's efficiency of serverless functions.

V. Summary and Preliminary Considerations

Due to the novelty of the DePIN paradigm, this paper identified a lack of studies exploring DePIN solutions focusing on provisioning of compute resources, especially when a serverless deployment model is considered. Furthermore, an overarching lack of direct comparisons with existing DePIN approaches with their centralized alternatives is present.

To address these limitations, this paper introduced the architecture of *Redacted*. Within this use case, the present work investigated the deployment of nodes within this DePIN and analyzed their service offering in terms of computational effectiveness and power efficiency. Furthermore, these results were contrasted with the results obtained from Cloud Service Providers (CSP). These initial results suggest that a mobile DePIN can perform comparatively to a local execution environment and potentially outperform remote CSPs, especially in terms of power efficiency.

However, in addition to acknowledging the limitations of the study (*i.e.,* the limited number of devices and the reliance on client-side data), future research will investigate the performance of additional workloads and consider the security of the platform against several threats.

Acknowledgment

This work has been partially supported by *(a)* the [redacted] and *(b)* the [redacted].

References

[1] J. Weinman, *Cloudonomics, + Website: The Business Value of Cloud Computing*, 1st ed. Wiley Publishing, 2012.

[2] A. Sants, "How Cloud Computing became a Global Monopoly," May, 2023, https://www.investorschronicle.co.uk/news/2023/05/09/how-cloud-computing-became-a-global-monopoly/.

[3] B. Edwards, S. Hofmeyr, and S. Forrest, "Hype and Heavy Tails: A Closer Look at Data Breaches," *Journal of Cybersecurity*, vol. 2, no. 1, pp. 3–14, 12 2016. [Online]. Available: https://doi.org/10.1093/cybsec/tyw003

[4] D. P. Mulligan, G. Petri, N. Spinale, G. Stockwell, and H. J. M. Vincent, "Confidential Computing - a Brave New World," in *2021 International Symposium on Secure and Private Execution Environment Design (SEED)*, 2021, pp. 132–138.

[5] [redacted], "[redacted]," December 2023.

[6] "Hivemapper," https://hivemapper.com/explorer.

[7] J. C. CFA and B. Chen, "DePIN: An Emerging Narrative," Binance Research, Tech. Rep., January 2024.

[8] X. Fan and L. Xu, "Towards a Rollup-Centric Scalable Architecture for Decentralized Physical Infrastructure Networks: A Position Paper," in *Proceedings of the Fifth ACM International Workshop on Blockchain-Enabled Networked Sensor Systems*, ser. BlockSys '23. New York, NY, USA: Association for Computing Machinery, 2024, p. 9–12. [Online]. Available: https://doi.org/10.1145/3628354.3629534

[9] M. C. Ballandies, H. Wang, A. C. C. Law, J. C. Yang, C. Gösken, and M. Andrew, "A Taxonomy for Blockchain-based Decentralized Physical Infrastructure Networks (DePIN)," *arXiv preprint 2309.16707*, 2023.

[10] X. Fan and L. Xu, "Towards a Rollup-Centric Scalable Architecture for Decentralized Physical Infrastructure Networks: A Position Paper," in *Proceedings of the Fifth ACM International Workshop on Blockchain-enabled Networked Sensor Systems*, 2023, pp. 9–12.

[11] D. Sarkar, "Generalised depin protocol: A framework for decentralized physical infrastructure networks," *arXiv preprint 2311.00551*, 2023.

[12] L. Icking, P. Felicio, S. Welde, J.-P. Doyen, R. Keenan, T. Nigg, and D. Ammann, "Onocoy: Enabling mass adoption of high precision gnss positioning using web3," 2023.

[13] J. Guo, Q. Guo, C. Mou, and J. Zhang, "A Mean Field Game Model of Staking Systemand A Reinforcement Learning Framework for Parameter Optimization," 2024.

[14] A. Datsenko, 2023, https://ideasoft.io/blog/what-are-decentralized-physical-infrastructure-networks-depin/.

[15] "Filecoin," https://filecoin.io.

[16] IoTeX, "Connecting the Real World to Web3," https://iotex.io/.

[17] IOTA, "An Open, Feeless Data and Value Transfer Protocol," https://www.iota.org/.

[18] The Helium Network, "People-Powered Networks," https://www.helium.com/.

[19] MXC Foundation, "Welcome to the MXC Foundation," https://www.mxc.org/.

[20] Streamr, "Decentralized Data Broadcasting," https://streamr.network/.

[21] J.-E. Ekberg, K. Kostiainen, and N. Asokan, "The untapped potential of trusted execution environments on mobile devices," *IEEE Security & Privacy*, vol. 12, no. 4, pp. 29–37, 2014.

[22] P. T. Maxime Rossi Bellom, Damiano Melotti, "2021: A Titan M Odyssey," 2021.

[23] J. Han, S. Huang, and Z. Zhong, "Trust in DeFi: An Empirical Study of the Decentralized Exchange," *Available at SSRN 3896461*, 2022.

[24] "Google Cloud Platform," https://cloud.google.com/.

[25] [redacted], "Submission to DePIN 2024 (ICBC Workshop))," https://github.com/d3pin/perf-analysis/releases/tag/submission.

[26] HPE, "AMD EPYC 7502P 2.5GHz 32-core 180W Processor Kit for HPE," 2024, https://buy.hpe.com/.

[27] Amazon, "Building Sustainable, Efficient, and Cost-Optimized Applications on AWS," 2023, https://aws.amazon.com/blogs/compute/building-sustainable-efficient-and-cost-optimized-applications-on-aws/.

All links above were last accessed on February 18th, 2024.