2023 IEEE International Conference on Blockchain and Cryptocurrency (ICBC) | 979-8-3503-1019-1/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICBC56567.2023.10174945

DeFaaS: Decentralized Function-as-a-Service for Emerging dApps and Web3

Rabimba Karanjai*,Lei Xu†, Nour Diallo*, Lin Chen‡, Weidong Shi*
*University Of Houston, TX, USA
{rkaranjai, ndiallo, wshi3}@uh.edu
†Kent State University, OH, USA , ‡ Texas Tech University, TX, USA
xuleimath@gmail.com, lin.chen@ttu.edu

***Abstract*—Function-as-a-service (FaaS) is an emerging computation architecture, which provides high scalability and flexibility. All the existing FaaS systems are owned and managed by a single cloud service provider. While this is not an issue for most existing enterprise applications, such character is not compatible with the decentralization principle of dApp/Web3 applications, more of which are being deployed in the cloud environment. Therefore, there is an urgent need to build a decentralized FaaS, which is managed by multiple cloud service providers and allows a decentralized application to take advantages of FaaS. In this research paper, we propose DeFaaS, a novel system for managing decentralized FaaS using blockchain technology and decentralized API management, where functions are executed on a distributed network of nodes by multi-cloud data centers, rather than on a centralized server. This allows for greater scalability and flexibility, as well as improved security and reliability.**

***Index Terms*—cloud, FaaS, blockchain, Web3, dApp, decentralization**

I. Introduction

The future of Web3 [1] and blockchain will likely be supported by both decentralized ICT (Information and Communications Technology) infrastructures like computation and storage resources contributed by volunteers [2] and managed ICT infrastructures like general-purpose computing resources provided by cloud service providers. In recent years, there are a number of projects proposing decentralized electronic marketplaces for computing resources, for instance, iEXEC [3], Golem [4], SONM [5], DRIVE (Distributed Resource Infrastructure for a Virtual Economy) [6], DFinity network [7], and many others.

In this work, we propose a transformative and one-of-its-kind decentralized infrastructure for Function-as-a-Service (FaaS). The infrastructure is unique in many aspects like support for dApp/Web3 applications to take advantage of the cloud resources, enabling multi-cloud FaaS for dApp/Web3 developers with a decentralized environment, a complete protocol stack based on open standards, and adherence to the principle of decentralization without relying on any centralized component. Furthermore, in this paper, we provide details of a concrete design using Hyperledge Besu and Open FaaS. Although we aim at multi-cloud data centers in this paper, the framework can be extended to encompass both managed and volunteer-contributed computing resources, a topic of future research.

II. Detailed Design of DeFaaS

In this section, we describe the details of the main components of the system.

A. Decentralized Scheduling and Load Balancing

DeFaaS utilizes a decentralized front-end for scheduling API calls, avoiding any centralized component. API calls can be made to any API gateway, with the management blockchain tracking reputation and supporting trust management of gateway nodes. API gateways need to be registered and may require a minimal stake. To prevent fragmentation, policies are enforced for gateway providers. Randomized load balancing is used to dispatch API calls across cloud data centers, with the power of two heuristic achieving good performance with low overhead [8], [9].

B. Decentralized Event Distribution

In DeFaaS, event distribution for dApps and Web3 applications is implemented using GossipSub [10], which is a message propagation system that relies on a mesh network structure and a score function to disseminate messages in a decentralized manner across multiple cloud service providers. GossipSub is adopted by several important blockchain systems including Filecoin and ETH2.0 Network. Function triggers can be implemented as a connector module in OpenFaaS, which is flexible enough to support multiple event trigger sources. The GossipSub connector maps topic-based events to the registered functions in OpenFaaS.

C. API Registration and Access Control

API providers register their API end-points on the management blockchain through a smart contract with a mapping to the provider's account. End-points can be public or private, with access control policies set for each end-point. Policies are stored in a map associated with the end-point and wallet address, and can be verified by querying the on-chain state. For public end-points, access verification is not needed.

TABLE I: Test application names & descriptions.

Name	Modules using IPFS for communication
User	User home directories
Web	Web/SQL server
Proj	Project directories
Ts	Terminal server
Src	Source control (Subversion)
Rsrch	Research projects (Carpooling dApp)
Mds	Media server (kodi)
Hm	Hardware monitoring (hwmon)

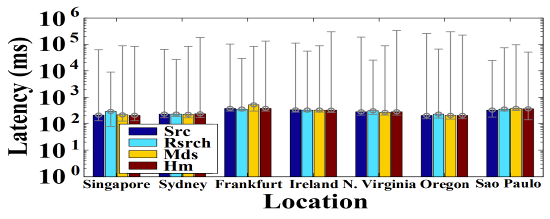

Fig. 1: Latency ranges for event messages at different locations.

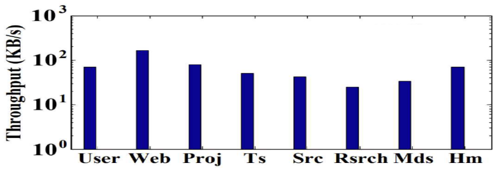

Fig. 2: Throughput of read operations over IPFS.

TABLE II: Performance of randomized load balancing (average queuing times in different scenarios: assume api call processed with one time unit, see text for explanation)

	6 data centers	8 data centers	10 data centers
Default implementation with randomly distributed calls	2.515	2.543	2.680
Default implementation with calls initially sent to one data center	2.574	2.574	2.826
Choice of two with randomly distributed calls	2.915	3.071	3.302
Choice of two with calls initially sent to one data center	3.029	3.199	3,657
Randomly distributed calls (no load balancing)	12.397	12.353	10.690

III. Implementation And Evaluation

A. Implementations

Our management blockchain is based on Hyperledger Besu [11] and we use containerized Open FaaS [12] on Kubernetes [13]. We require extensions to Open FaaS: a connector for external event invocation via decentralized IPFS, an interoperability plugin for authentication verified against the management blockchain, and randomized load balancing using a modified Power of Two Choices. Data sharing between cloud data centers uses IPFS and Web3 storage. Cross-chain transactions are supported by Besu using existing cross-chain contracts and bridge components, such as SOFIE Interledger bridge [14]. API call logging is done through IPFS.

B. Evaluation

a) **Multi-cloud event dissemination:** Multi-cloud event distribution performance is evaluated. We first present the latency with different applications from remote nodes in Fig. 1. For small messages (Fig. 1), the latency varies significantly across different remote nodes. In particular, the maximum latency can be much higher than the median value.

b) **Effect of IPFS read operations:** We evaluate downloading performance across clouds by measuring latency and throughput of read operations. Remote node locations do not significantly impact read latency, likely due to the high variance in latency for small read requests. Local node resolving and downloading operations significantly affect I/O performance for remote read operations.

c) **Randomized load balancing:** To evaluate the performance of randomized load balancing, we have developed a simulator for experiments. The simulator models the API gateways and routing of API calls to multi-cloud data centers. The simulator allows us to explore load balancing performance of different heuristics: (i) a default design based on the standard power of two choices; (ii) a simplified heuristic called choice of two; and (iii) a baseline of no randomized load balancing. In the default design, an API gateway randomly selects another cloud data center. If the receiving data center is busier, the call is forwarded to the randomly picked data center. In the choice of two heuristic, the receiving cloud's load is evaluated, and if it's overloaded, the call is forwarded to a randomly picked data center. In the baseline, there's no randomized load balancing.

We simulated 6, 8, and 10 data centers, with results averaged over five simulation runs of 10,000 API calls each. The default implementation of randomized load balancing achieves the best performance in terms of average queue times, followed by the choice of two option, which only performs slightly worse. The worst average queue times are observed without randomized load balancing. While the second option has slightly longer average queue times, it may still be a good candidate due to its simplicity and lightweight implementation. The heuristic and simulator can be modified to use multi-dimensional utility that combines cost, decentralization, and performance.

IV. Conclusions

FaaS is a new computation paradigm, and offers high flexible and scalable computation capability. While it is feasible for an dApp/Web3 application to take advantage of the existing FaaS, it greatly impacts the decentralization character as most FaaS is owned and managed by a single cloud service provider. We propose DeFaaS in this work, which utilizes blockchain technology to coordinate FaaS systems of multiple cloud service providers. Detailed design of the key components are provided, and preliminary prototypes and experiments demonstrate the practicability of the system to support dApps/Web3 applications.

References

[1] J. Bambacht and J. Pouwelse, “Web3: A decentralized societal infrastructure for identity, trust, money, and data,” *CoRR*, vol.

abs/2203.00398, 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2203.00398

[2] M. Nouman Durrani and J. A. Shamsi, "Review: Volunteer computing: Requirements, challenges, and solutions," *J. Netw. Comput. Appl.*, vol. 39, p. 369–380, mar 2014.

[3] iEXEC, "Blockchain-based decentralized cloud computing," https://iex.ec/wp-content/uploads/pdf/iExecWPv3.0-English.pdf, 2017.

[4] G. Network, "Golem network: Online white paper," https://golem.network/doc/Golemwhitepaper.pdf, 2016.

[5] SONM, "Supercomputer organized by network mining," https://whitepaper.io/document/326/sonm-whitepaper, 2017.

[6] K. Chard and K. Bubendorfer, "Co-operative resource allocation: Building an open cloud market using shared infrastructure," *IEEE Transactions on Cloud Computing*, vol. 7, no. 1, pp. 183–195, 2019.

[7] T. Hanke, M. Movahedi, and D. Williams, "Dfinity technology overview series, consensus system," 2018. [Online]. Available: https://arxiv.org/abs/1805.04548

[8] M. Mitzenmacher, "The power of two choices in randomized load balancing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 12, no. 10, pp. 1094–1104, 2001.

[9] ——, "The power of two choices in randomized load balancing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 12, no. 10, pp. 1094–1104, 2001.

[10] D. Vyzovitis, Y. Napora, D. McCormick, D. Dias, and Y. Psaras, "Gossipsub: Attack-resilient message propagation in the filecoin and ETH2.0 networks," *CoRR*, vol. abs/2007.02754, 2020. [Online]. Available: https://arxiv.org/abs/2007.02754

[11] Besu, "Hyperledger besu," https://besu.hyperledger.org/, accessed: 04/20/2021.

[12] OpenFaaS, "Openfaas - serverless functions made simple," https://github.com/openfaas/faas, accessed: 09/11/2022.

[13] Kubeless, "Kubernetes native serverless framework," https://github.com/kubeless/kubeless, accessed: 10/17/2022.

[14] Github, "Sofie interledger repository," https://github:com/SOFIE-project/Interledger.