Decentralized Function as a Service

*Department of Software Engineering, Sri Lanka Institute of Information Technology (SLIIT), Malabe, Sri Lanka.

1st Lakmal Rupasinghe *lakmal.r@sliit.lk*

2nd R.G.D Nayomal *dinushankanrg@gmail.com*

3rd S.K.N.U Tissera *nuwan.tissera@my.sliit.lk*

4th T.H.A.K Silva *achala.kavinda@my.sliit.lk*

5th A.T Nimansa *tharindi.nimansa@my.sliit.lk*

***Abstract*—The proposed system, "Orb" is a decentralized Function as a Service provider that replaces centralized, single-authority FaaS, which utilizes the idle computing power and addresses the drawbacks of serverless computing. The serverless architecture suffers from drawbacks such as Vendor control, multi tenancy problems, vendor lock-in, security concerns difficulty of managing of granular functions and architectural complexity. The proposed system resolves the vendor control, multi tenancy, vendor lock-in, security concerns and managing of granular functions by decentralizing the granular functions across peers. A granular function refers to an atomic method/function with a single responsibility (such as addition, image recognition, IoT connector) hosted in the network. The "Orb" is a peer to peer network consisting of nodes and supernodes. The node discovery of the network is done by adhering to "Satoshi Client Node Discovery". The user could sign up to the network as a node or a supernode by installing client or the server software provided. Node provides the functionality of implementing, deploying, hosting functions. Supernode keeps a track of nodes, supernodes and hosted granular functions. The user could deploy a function to the network through the client software. For deployment, user will be charged in cryptocurrency. The user will be paid in cryptocurrency considering the no of requests served and uptime. The payments model of "Orb" depends on ether smart contracts. The confidentiality of a deployed function is achieved through techniques such as public key private key encryption, proper packaging and containerization.**

***Index Terms*—Cryptocurrency, DFaaS, Decentralization**

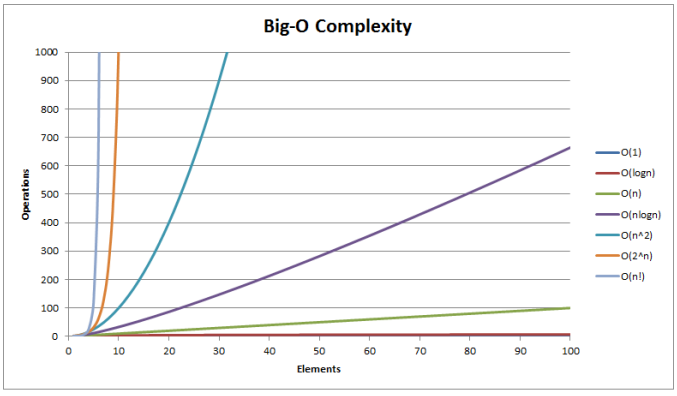
I. Introduction

Serverless architectures refer to applications that depend on 3rd party services (knows as Backend as a Service or "BaaS") or on custom code that's run in ephemeral containers (Function as a Service or "FaaS"), the best-known vendor host of which currently is AWS (Amazon Web Services) Lambda (FaaS provider). The name "serverless computing" is used as the end user doesn't have to code the backend [15] to run. Serverless code can be used together with traditional architectures, such as microservices [16]. For example, part of a web application could be written as microservices and another part could be written as serverless code. Alternatively, an application could be completely serverless if it's written only using granular functions without the backend [15]. A granular function refers to an atomic method/function with a single responsibility (such as addition, image recognition, IoT connector) hosted in the network. FaaS helps developers to code the functions without paying attention to the server infrastructure. But serverless architecture has a notable set of drawbacks since server infrastructure is maintained by the vendor. Vendor control, multi tenancy problems, vendor lock-in, security concerns, lack of debugging and monitoring tools, difficulty of managing of granular functions and architectural complexity as some of the drawbacks. The proposed decentralized Function as a Service solution, "Orb" is a unique research area which changes the definition of serverless computing. Today most systems are centralized and has a single-authority. Large scale cloud services providers will decide how the data is manipulated and controlled along with the cost. The proposed system will benefit the users by giving control of their and power of information back to them. Idle computing power is available freely over the world. But there is no proper mechanism of utilizing the computing power to perform computations. By allowing each node to run function/functions and exposing them could be used to utilize this computing power. The "Orb" focuses on the following challenges.

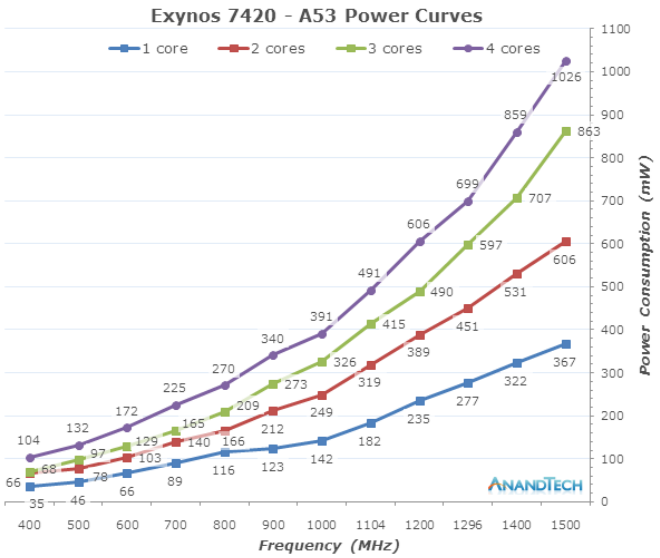
- Decentralized Service Deployment
- Network Infrastructure
- Function as a Service Provider
- Payments Model
- Security

II. Methodology

A. Decentralized service deployment

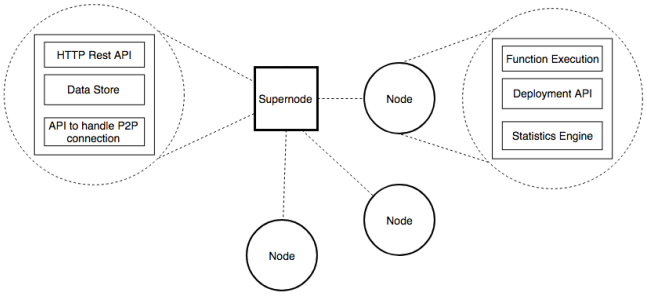


After developing a certain service (API, Lambda function), the user needs to host it to use it. The cloud cost increases with the increment of algorithm complexity and computational power.



The developers or the owners of the app might tend to host it on centralized cloud computing platforms like Amazon Web Services. There can be outages [11] since all centralized systems comes with a single point of failure [12]. The most recent outage being the "AWS S3" outage [1] where 1000s of websites suffered a downtime. The proposed solution can be advantageous since the single point of failure [12] is eliminated through decentralization. The existing cloud computing IaaS such as AWS EC2 and GCP Compute engine, the user is charged for the storage, memory and the number of cores of the CPU and more importantly, the code that's deployed to the above-mentioned providers are owned by the vendors. Decentralizing functions will result in elimination of the single authority. Since "Orb utilizes only the idle and available computers to host the functions, the computing cost is reduced. "Orb" will monitor and pay in terms of cryptocurrency according to the the usage of computational power in terms of uptime and number of requests.

B. Network Infrastructure



The proposed system will use network of nodes and super nodes.

C. Node

Node represent a personal computing device with the Orb client software installed. The responsibilities of the node include deploying a function to the Orb, keeping a track of other nodes and super nodes, ability to download an available function from the supernode and hosting it on the node. Hosted function on the client machine will generate income in terms of Crypto Currency for the user. The user can subscribe to a payments model when deploying an application. The client nodes will have a wallet to secure the payments received. Node will consist of the Multiple replicas of the same function will be deployed in different nodes and a supernode will automatically deploy the function when there are no active nodes available to ensure that the functions deployed are available.

D. Supernode

Super node represents a node with the Orb server installed. Keeping a track of nodes and super nodes, API endpoint generator to provide a unique endpoint to consume the deployed function, REST API to communicate with the outside world for data analytics are the main functionalities of a super node. A user can choose to be a super node. But in order to be a super node, it must be on a public network with a public domain name assigned to it.

E. Function as a service provider

The user doesn't have to configure a server resources such as storage, memory. This is because, the Orb calculates the complexity of the function by executing it. Then the function is only distributed across the nodes which can only serve the computational requirements of the functions. The user deploys a function and an endpoint will be generated and sent to the user. Even though configuration overhead is eliminated. The functions cannot be edited or removed easily due to the decentralized nature of the system.

F. Bootstrapping Node

Bootstrapper node provides initial configurations for newly joining Nodes and Trackers. It helps to establish initial connection with existing Nodes and Trackers in the system. When a node or supernode wants to connect with the system, it establishes an initial connection with bootstrapper node. Then bootstrapper provides details about supernodes. Then node or supernode can successfully establish the connection with the system.

G. Payment model

Decentralizing server space to host a function makes services provided more reliable. Decentralized Functions as a service or in one term, "Orb" grant the user the privilege of enjoying a reliable secure and cost-effective service. The purpose of "Rewarding system" in orb is simply to provide it's users who act as function seeders, owners and requesters a payment model which acts in a logical manner to provide fair rewards to all.Payment Model based on requests served by a node. The functionalities our rewarding system should possesses includes,

I. Finding out the possibility of paying the nodes which hosts functions based on served requests The function owner

initially hosts his own function and if some user is interested in hosting it, he can seed it to be used by requester. When requester, request for a specific function the function owner and the seeder both gets rewarded as per the requests for a pre-defined ratio

II. Finding out a mechanism to validate the requests served by a mechanism between the origin of the request and the requests served Validating the requests mainly focuses on blocking, function owner or seeder to send requests to it's own function and corrupt the rewarding system. For the above-mentioned function finding out a secure mechanism to prevent DoS attacks which lets a node earns money by sending pings should be done Main Goals of the rewarding system

I. Provide a payment model • Pay the function owner • Pay function seeders • Charge from function requesters

II. Calculate the complexity of the functions Calculating the complexity of the functions is important to give a value to each function. There can be many methods to calculate it such as • Function execution time • Lines of code • Size of the function

III. Calculate the market value of the function taking the complexity of the function as the base Since the functions are distributed we should use one of the above-mentioned methods which will give the approximate comparisons of the complexities and let us use the value generated to get a price.

IV. Automatically distribute the rewards There should be a way to identify the connected nodes, and filter the seeders, owners separately. When the function is requested by another node the number of requests for the function should be counted and then the rewards should be distributed. For this we should create a program which will act as an intermediary contract to autonomously charge the prices and grant the rewards

H. Security

Orb is invulnerable to DDoS attacks due to the decentralization natively. Only an executable of the deployed function will be deployed (without the source code) as a security mechanism. To ensure that the decentralized functions are not changed, the super nodes will automatically verify by comparing a dynamically generated hash and the existing hash, each time a node with a specific function comes live. To ensure reliability and to avoid single point of failure, there will be a chain of super nodes connected to each other.

I. Deployment and distribution of functions in the network

When the user starts the Orb client software, subscribe to the nearest available supernode. When the user uploads a function through the client, the client will ask the user to subscribe for a payments plan (an Ethereum smart contract). Then function will run on the same client machine first. Then function source code will be run through a hash function and hashed. This will generate an initial hash which can be used to identify the function. Also, a key pair will be generated for a function. Then the function will be converted to an executable file. Then function will be send to the super node. The supernode will initially run the function, generates a URL and send to the deployed user. Then supernode will look for the live nodes list and sends a notification about the availability of the function. The nodes can now choose to run the function or not. This way, the functions will be distributed throughout the network.

J. Updating and deprecation of the function

The deployed user can initiate a function deprecation or an update. Then this request will be propagated through the network. The updating and deprecation process will be slow since the system is decentralized. Orb will simply minimize the risk of single point of failure since multiple nodes hold the responsibility of function availability. In any system usually, trade-offs are done considering efficiency, reliability security scalability etc. A decentralized index approach tends to be more robust (no single point of failure), but it is usually tricky to make it as efficient as a centralized approach. Scaling will not cost extra money anymore. In real world if the system/software becomes bigger the servers should scale horizontally or vertically. We can scale our own machines or use the most common method of using a cloud services platform but both methods are costly using Orb the expenses for scaling can be reduced and can be controlled. The function owner can monitor the usage and if that function exceeds request limit the owner of the function must pay all the hosted nodes through an Ethereum contract. When the user puts the requested amount of ether to the contract, will be distributed among the nodes based on the involvement of nodes. Free server spaces can be utilized to a worthy cause Starting from a smart phone up to a server farm anyone interested can provide server space and earn extra rewards. This will reduce the cost for scaling which is beneficial for the function owner and reward the nodes who provide server space.

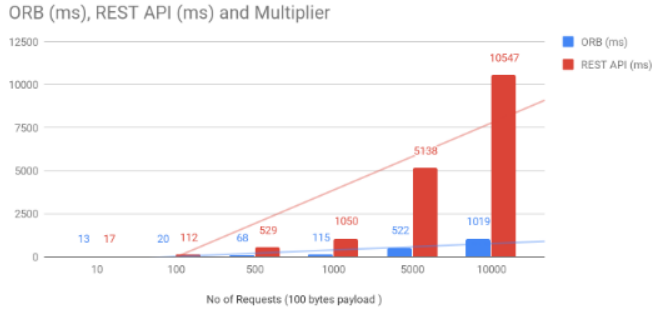
III. Results

Orb uses sockets as a key component in its infrastructure to communicate between the nodes and super nodes. In order to test the performance, the following function was hosted on orb and as a REST API.

```
function add(number1, number2)  {
return number1 + number2;
}
```

The Orb network had 2 nodes and a super node which runs on a t2.micro AWS instance routing the requests. The function was hosted on two laptop computers which had the following specs, 2.5GHz clock speed, 16GB of Ram and 512GB SSD. The Rest api was setup using nginx reverse proxy in a t2.micro instance hosting the above mentioned function. The results were as follows.

Constant payload with changing number of messages to the above function.

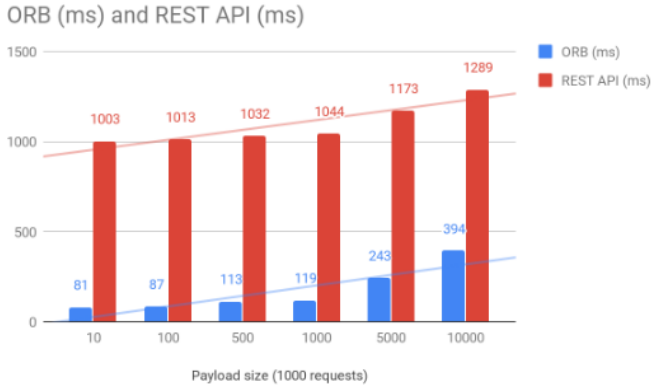


No of Requests (100 bytes payload)	ORB (ms)	REST API (ms)	Multiplier
10	13	17	1.31
100	20	112	5.60
500	68	529	7.78
1000	115	1050	9.13
5000	522	5138	9.93
10000	1019	10547	10.35

Constant number of messages with incremental number of requests to the above function

Using the above results, it's concluded that ORB is a more efficient than RESTful HTTP calls. But the purpose of ORB is to create a community driven platform to host and consume functions in a decentralized manner. *Lower is better

Payload size (1000 requests)	ORB (ms)	REST API (ms)	Multiplier
10	81	1003	12.38
100	87	1013	11.64
500	113	1032	9.13
1000	119	1044	8.77
5000	243	1173	4.83
10000	394	1289	3.27



IV. Conclusion and Future Work

In this paper, we have presented ORB, Decentralized Functions as a Service provider implemented on top of sockets. It supports horizontal scaling and invulnerable to DDoS attacks with the incremental number of nodes and super nodes.From the implementation, we conclude that ORB has the ability to use commodity hardware in a decentralized manner to provide functions as a service. From the comparison of centralized HTTP based RESTful APIs, sockets are much faster. With the implementation of a decentralized rewarding platform to support the function seeders, hosters and consumers, Orb has the capability of utilizing idle computing power of commodity hardware. In future, Orb will be implemented with more fault tolerant features. The performance of the system can be increased by introducing decentralized caching mechanisms for certain types of hosted functions and optimizing the function call routing and distribution algorithms.

Acknowledgment

This is to acknowledge the efforts of one and all to make this research a success. Thank you everyone for dedication and hard work. We would also like to show our gratitude to our supervisor Mr. Lakmal Rupasinghe (Senior lecturer - faculty of computing) for sharing his pearls of wisdom with us during this research

References

[1] D. Etherington, "Amazon AWS S3 outage is breaking things for a lot of websites and apps", TechCrunch, 2018. [Online]. Available: https://techcrunch.com/2017/02/28/amazon-aws-s3-outage-is-breaking-things-for-a-lot-of-websites-and-apps/. [Accessed: 14- Jan- 2018].

[2] M. Roberts, "Serverless Architectures", martinfowler.com, 2018. [Online]. Available: https://martinfowler.com/articles/serverless.html#unpacking-faas. [Accessed: 14- Jan- 2018].

[3] "Serverless computing", En.wikipedia.org, 2018. [Online]. Available:https://en.wikipedia.org/wiki/Serverless_computing [Accessed: 14- Jan- 2018].

[4]]"What is Ether", Ethereum.org, 2018. [Online]. Available: https://www.ethereum.org/ether. [Accessed: 14- Jan- 2018].

[5] R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.

[6] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].

[7] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.

[8] "How Do Ethereum Smart Contracts Work? - CoinDesk", CoinDesk, 2018. [Online]. Available: https://www.coindesk.com/information/ethereum-smart-contracts-work/. [Accessed: 14- Jan- 2018].

[9] "ConsenSys – Medium", Medium.com, 2018. [Online]. Available: https://medium.com/@ConsenSys. [Accessed: 14- Jan- 2018].

[10]]"What Is AWS Lambda? - AWS Lambda", Docs.aws.amazon.com, 2018. [Online]. Available: https://docs.aws.amazon.com/lambda/latest/dg/welcome.html. [Accessed: 14- Jan- 2018].

[11] Steemit.com, 2018. [Online].Available: https://steemit.com/faq.html#What_is_Steemit_com. [Accessed: 14- Jan- 2018].

[12] "AWS Rates Highest on Cloud Reliability", EnterpriseTech, 2018. [Online]. Available: https://www.enterprisetech.com/2015/01/06/aws-rates-highest-cloud-reliability/. [Accessed: 14- Jan- 2018].

[13]]"The AWS Outage: The Problem with Internet Centralization — Mondo", Mondo, 2018. [Online]. Available: https://www.mondo.com/aws-outage-internet-centralization-problem/. [Accessed: 14- Jan- 2018].

[14]]"Single Point of Failure – What is it? Why it Matters. . . :", Renovodata.com, 2018. [Online]. Available: http://www.renovodata.com/blog/2015/06/10/single-point-of-failure. [Accessed: 14- Jan- 2018].

[15] "www.ey.com", Ey.com, 2018. [Online]. Available: http://www.ey.com/Publication/vwLUAssets/EY-implementing-blockchains-and-distributed-infrastructure/$FILE/EY-implementing-blockchains-and-distributed-infrastructure.pdf [Accessed: 14- Jan- 2018].

[16] P. Labs, ”IPFS is the Distributed Web”, IPFS, 2018. [Online]. Available: https://ipfs.io/. [Accessed: 14- Jan- 2018].
[17] ”Storj - Decentralized Cloud Storage”, Storj - Decentralized Cloud Storage, 2018. [Online]. Available: https://storj.io. [Accessed: 14- Jan- 2018].
[18] A. Frumusanu, ”The Samsung Exynos 7420 Deep Dive - Inside A Modern 14nm SoC”, Anandtech.com, 2018. [Online]. Available:https: //www.anandtech.com/show/9330/exynos-7420-deep-dive/5. [Accessed: 11- Feb- 2018]
[19] ”Part II - Determining Complexity of Algorithms”, 2018. [Online]. Available: http://arturmeyster.com/time-complexity/#.WoA96aiWbDc. [Accessed: 11- Feb- 2018]