Title of the Invention

Decentralized Functions as a Service

Background of the Invention

Nowadays most systems are centralized and has a single-authority. Due to the overhead that is attached with the deployment of a centralized system, users have turned towards light weight Serverless functions. Serverless functions helps the users to avoid the creation of large web application programming interfaces and focus more towards the functionality. Serverless functions follows an event-driven setup where the configuration and handling of servers is automatically handled by the platform, hence the name “Serverless”.

Serverless architectures refer to applications that depend on 3rd party services (Known as Backend as a Service or “BaaS”) or on custom code that’s run in ephemeral containers (Function as a Service or “FaaS”). AWS (Amazon Web Services) offers a serverless computing platform known as Lambda (FaaS provider) which is used widely.

Serverless code can be used together with traditional architectures, such as microservices. 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. But Serverless Architecture has a notable set of drawbacks. Vendor control, multi tenancy problems, vendor lock-in, security concerns, difficulty of managing a large number of granular functions are some of the drawbacks.

Accordingly, if there is a possible way of avoiding the above-mentioned drawbacks, using Serverless functions, will be more solid. Although there are many service providers which provides server space to deploy applications, most of them works in a centralized manner. But there are no solutions which provides these services in a decentralized manner. As for the Function as a Service is concerned, there’s no proper way or existing systems which use the concept of decentralized FaaS as of today. This invention offers a solution to the existing vendor control, multi tenancy, vendor lock-in, security concerns and managing of granular by decentralizing granular functions. A granular function refers to an atomic method/function with a single responsibility (such as addition, image recognition, IoT connector) hosted in the network.

Summary of the invention

The invention decentralizes the granular functions throughout a peer to peer network in order to provide a decentralized Function as a Service. This invention describes a method for the user to log into the super node network using the client software which is installed on a node where the node is identified by a unique identifier. Then the user has the privilege to use the inline code editor built into the client software to create a function, obfuscate the code, package the function and run the function on the local node. Upon the execution of the function in the local node, the complexity of the function is calculated. This complexity is used to calculate the cost of the Reward for the function in order to deploy the function to the network. The user must reward in order to let the other users interact with the function. The other nodes are notified about the presence of the function though a set of specialized super nodes in the network. According to the notifications coming from the network, a node may choose to consume the hosted function or seed the function. Decentralized Rewarding system integrated to the network will reward the users in terms of their actions such as deploying functions, seeding functions and consuming functions

Brief Description of the Figures

Figure 1 – High level Architecture

1 - Tracker is usually a super node identified by the Orb. It has three main sections as HTTP Rest API, data store and API to handle peer to peer connection

2 - Node is a normal user device which the client software has been installed. It has the running function, deployment API and statistic engine in it. A function owner can write a function obfuscate and deploy the function or a function requester can choose a function, subscribe and consume it. Additionally, a function seeder can choose a function and seed it.

Figure 2 – Getting registered to Orb

3 - Orb is the collection of nodes and Trackers (supernodes) as explained in figure 1

4 - Initially when a node is registered and installed the client software, it will be rewarded with a predefined amount of currency which is native to the system.

5 - When the reward is received the function owner can deploy the function into orb by spending an amount as per the complexity of the function

6 - E wallet is the application where all the transactions are tracked and securely keep the user's earned rewards

Figure 3 – function deployment, function request and function seeding

7 - After a function is deployed in Orb, node who is interested in that function can choose the function, subscribe and consume it. for consuming the function, the requester must pay as per the number of requests

8 - When other nodes are requesting for the function, function owner gets rewarded as per the request count

9 - Function seeder is getting rewarded for the functions he seeded according to the requested he served.

Detailed Descriptions of the Invention

Invention consist of 3 major components.

1. Nodes
2. Super Nodes
3. Decentralized Rewarding System

1. Nodes

Node represent a personal computing device with the client software installed. The responsibilities of the node are as follows
Deploying a function to the network.
Keeping a track of peers and super nodes - Using DHT and super nodes servers
Provide a URL to call a deployed function by the user
Able to run an available function - By running an available function on the client machine will generate income based on the decentralized Rewarding System
Subscribe to a Rewarding model when deploying an application
A client side wallet to secure the Rewards received

Node will consist of the following proposed mechanisms to ensure that the functions deployed are available.
Multiple replicas of the same function will be deployed in different nodes
A tracker will automatically deploy the particular function when there are no active peers available.

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 peer comes live.

To ensure reliability and to avoid single point of failure, there will be a chain of super nodes connected to each other.

2. Super node

Super node represents a node with the server installed. The responsibilities of the super node are as follows
Keeping a track of peers and super nodes
API endpoint generator which generates global endpoints for deployed functions
REST (Representational State Transfer) API to communicate with the outside world for data analytics
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 domain name assigned to it.

3. Decentralized Rewarding System

In the proposed system, the initial deployment will cost the user in terms of Rewarding mechanism. A particular user can deploy a function. The decentralized network will decide the deployment process. Also the network will monitor the usage of the function. If that function exceeds particular request limit the owner of the function has to pay all the hosted nodes through the Rewarding mechanism. When the user puts the requested amount of Rewards to the network, will be distributed among the nodes based on the involvement of nodes. The nodes will be rewarded in terms of the initially allocated amount

Deployment and distribution of functions in the network works as follows. When the client starts the client software, it will send a request to the super node and the super node will start to keep a track of the particular node.

When the user uploads a function through the client, the client will ask the user to subscribe for a rewarding plan. 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 particular function. Then the function will be converted to a 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.

The deployed user can initiate a function deprecation or a update. Then this request will be propagated through the network. The updating and deprecation process will be slow since the system is decentralized.

Abstract of Disclosure

Orb provides distinctive a Function as a Service(FaaS) through a network of interconnected nodes. It will allow the user to have a secure and cost-efficient method of hosting serverless functions. Node can be of two kinds as, supernodes and client nodes. Orb is the combination of decentralized super nodes and nodes with an embedded rewarding system. Super nodes act as trackers and normal nodes can be formed by installing the client software in a personal computer. User can host a function in a node. The can use a function which is already hosted. Users can choose to host functions deployed by other users. Consuming a hosted function can be done by sending a request with relevant parameters to the Orb. Furthermore, getting rewarded for hosting functions and deployment is a key feature built into the Orb.

Claims

1. Serving functions over decentralized nodes which can be consumed through another decentralized node.

Initially when the system boots up, there is a set of known super nodes. When user install client application client application know what are super node are existing in network. These nodes are connecting socket streams. Which allows platform to communicate bi-directionally. When a client gets registered in the system, supernode identify it as a node. Node is identified using unique Orb id. Even after every node addition, supernode populates its data which is saved in a Distributed hash table (DHT) throughout the network. This makes the transparency between supernodes like nodes feel it as a central computer. When a super node or node are connecting each other, they are exchanging. critical information to maintain connection between them and it is important exchange parameter to bridge gap between public to private network gap. This information must store in both client node peer and super node peers. and, it must be persistent and efficient way to storing, retrieve this data. store data can be used to re-initiate connection if failure has occurred.

2. Generation of a network wide notification about a new function deployment where the client nodes can host the same function by acknowledging the notification

Orb platform allowed its client to host their function in distributed environment and auto scale up to provide an efficient function hosting platform to its user. In this auto scaling we need to find out proper mechanism to mirror a function in one client node to another client node with in distributed system. this will provide reroute traffic between nodes once a node is busy to provide response. When a new function is deployed the whole network will get notified then the node who have the potential to host the function can acknowledge it and seed it

3. Rewarding the function owner based on the function consumption by client nodes through a decentralized rewarding system

Function owner creates a useful function and allow other peers in the system to use it. the consumption of the function can be measured with the number of requests to the function. According to the number of requests the function owner will get a reward in the form of an electronic currency which is native to the Orb. This rewarding method is decentralized hence no central authority who will govern the transactions happening. But the Orb system will ensure that the correct parties will get benefitted and rewards are fairly distributed

4. Rewarding the function hosting nodes through the decentralized rewarding system

Function hosting node will provide their resources as a server space to a function deployed by some other node. This seeding process will help to keep the network live without downtimes. The inefficiently used resources can be used for a useful cause letting the function hosting nodes to acquire extra income. The count of the requests for the function will be taken as a measure to distribute the rewards to all the function hosted nodes.

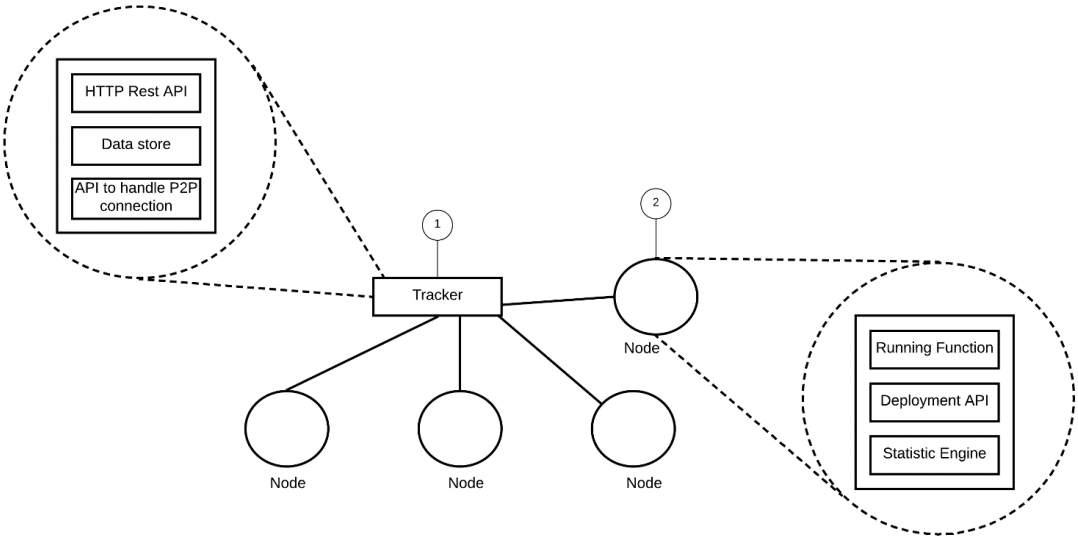

Figure 1- High-level Architecture

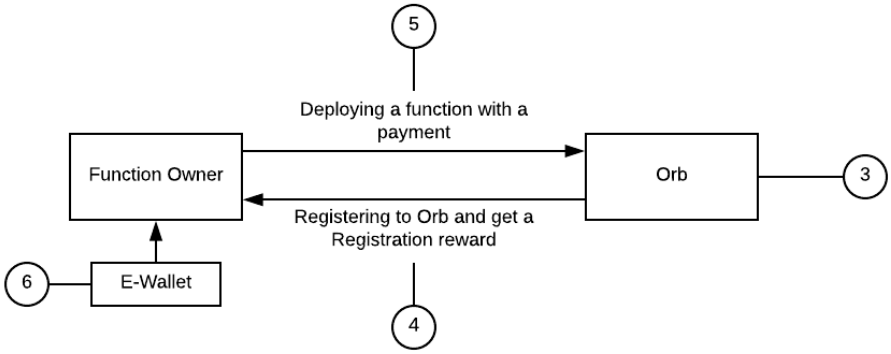

Figure 2- Rewarding System - part 1

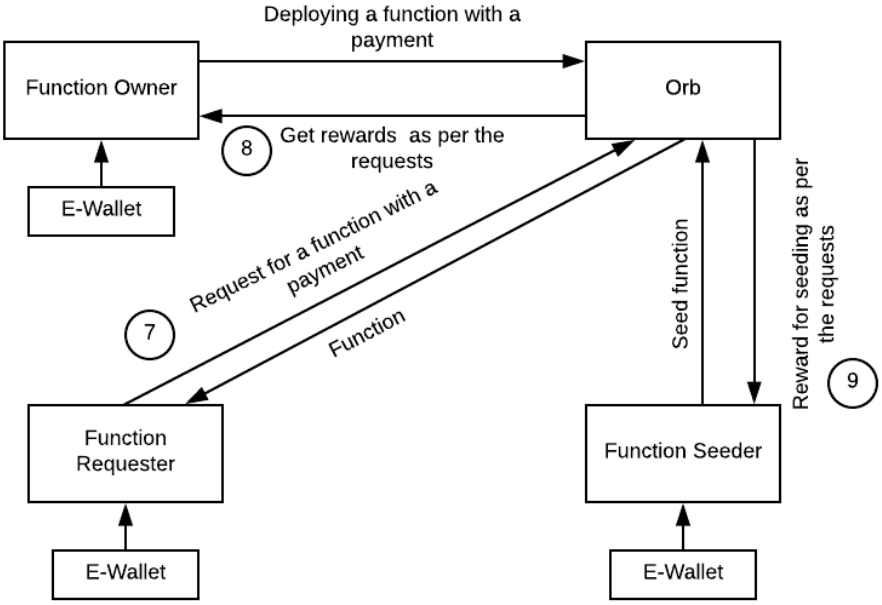

Figure 3- Rewarding System - part 2