Sri Lanka Institute of Information Technology

PROJECT REGISTRATION FORM

(This form should be completed and submitted on or before 3.00 PM, Friday 3rd March, 2017)

The purpose of this form is to allow final year students of the B.Sc. (Hon) degree program to enlist in the final year project group. Enlisting in a project entails specifying the project title and the details of four members in the group, the internal supervisor (compulsory), external supervisor (may be from the industry) and indicating a brief description of the project. The description of the project entered on this form will not be considered as the formal project proposal. It should however indicate the scope of the project and provide the main potential outcome.

PROJECT TITLE	Decentralized Functions as a Service
RESEARCH GROUP	Data Communication
PROJECT NUMBER	(will be assigned by the lecture in charge)

PROJECT GROUP MEMBER DETAILS: (Please start with group leader's details)

	STUDENT NAME	STUDENT NO.	CONTACT NO.	EMAIL ADDRESS
1	R.G.D Nayomal (GROUP LEADER)	IT15027948	071 157 0067	dinushankanrg@gmail.com
2	S.K.N.U Tissera	IT15120212	077 585 6964	nuwan.tissera@my.sliit.lk
3	T.H.A.K Silva	IT15136916	077 358 4572	achala.kavinda@my.sliit.lk
4	A.T.Nimansa	IT15419910	076 858 5463	tharindi.nimansa@my.sliit.lk

SUPERVISOR
Mr. Lakmal Rupasinghe

Name	Signature	Date

CO-SUPERVISOR (will be assigned by the Supervisor, if necessary)

Name	Signature	Date

EXTERNAL SUPERVISOR (if any, may be from the industry)

Name	Affiliation	Contact Address	Contact Numbers	Signature/Date

ACCEPTANCE BY CDAP MEMBER

Name	Signature	Date

PROJECT DETAILS

Brief Description of your Research Problem:

The proposed decentralized Function as a Service solution is a unique research area which changes the definition of serverless computing. Today most systems are centralized and has a single-authority. The data is owned by around 150 large scale companies. These companies will decide how the data is manipulated, charged and controlled. The proposed system will benefit the users by giving control of their and power of information back to them.

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 to run. 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. 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 system resolves the vendor control, multi tenancy, vendor lock-in, security concerns and managing of granular functions by decentralizing the granular functions throughout a peer to peer network in order to provide a decentralized FaaS. A granular function refers to an atomic method/function with a single responsibility (such as addition, image recognition, IoT connector) hosted in the network.

Description of the Solution:

Introduction to Orb

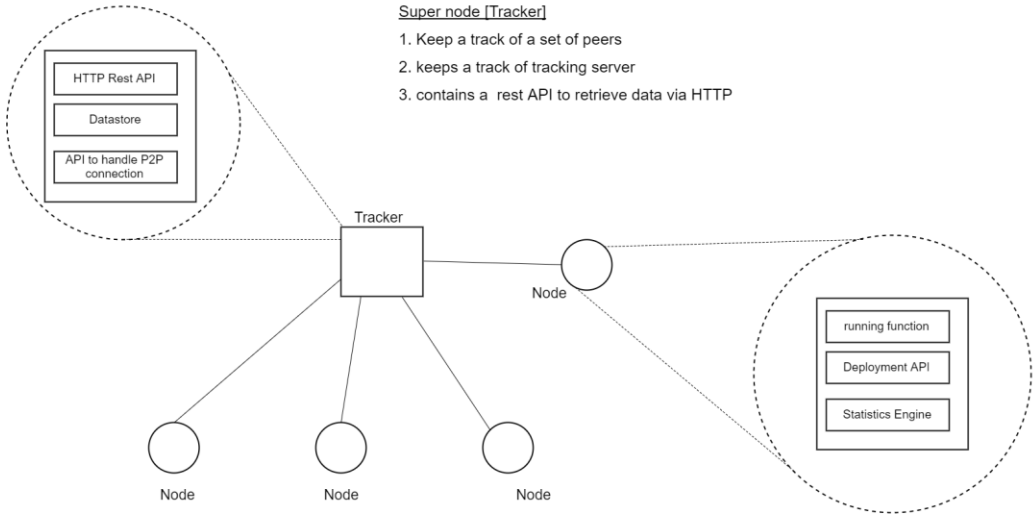
Orb is a decentralized function as a service provider. It consists of 3 major components.

1. Nodes

2. Super Nodes [Trackers]

3. Payments Manager

High level Architecture



Super node [Tracker]

1. Keep a track of a set of peers
2. keeps a track of tracking server
3. contains a rest API to retrieve data via HTTP

node

1. Keep a track of a set of peers
2. Able to deploy a function to the network
3. Able to call the deployed function
4. contains a list of hardcoded always running trackers to initiate the network
5. Able to run and deploy function

Nodes

Node represent a personal computing device with the Orb client software installed. The responsibilities of the node are as follows

1. Deploying a function to the network.

2. Keeping a track of peers and super nodes - Using DHT and super nodes servers

3. Provide a URL to call a deployed function by the user

4. Able to run an available function - By running an available function on the client machine will generate income in terms of Ethereum for the user.

5. Subscribe to a payments model when deploying an application

6. A wallet to secure the payments received

Node will consist of the following proposed mechanisms to ensure that the functions deployed are available.

1. Multiple replicas of the same function will be deployed in different nodes

2. 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.

Super Nodes [Trackers]

Super node represents a node with the Orb server installed. The responsibilities of the super node are as follows

1. Keeping a track of peers and super nodes

2. API endpoint generator which generates global endpoints for deployed functions

3. 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.

Payments Manager

Owner

In the proposed system, the initial deployment will cost the user in terms of ether [5]. 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 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.

Nodes

The nodes will get paid in terms of ether according to the contribution to the network.

Main expected outcomes of the project:

General Functionality

1. Deployment and distribution of functions in the network

When the client starts the Orb 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 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 particular function. Then the function will be converted to an executable file. Then function will be send to the super node. The super node will initially run the function, generates a URL and send to the deployed user. Then super node 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.

2. 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.

WORKLOAD ALLOCATION (Please provide a brief description about the workload allocation)

R.G.D Nayomal	....

- Finding an efficient mechanism to explore the possible ways of bi-directional communication between private and public networks.

- Main objectives
  - To decrease latency of data communication between the nodes.
  - To improve communication between nodes in the same network efficient
  - Determining secure ways to store functions in nodes

- Finding out a mechanism to revert / edit a function deployed in the network
- Finding out the possibility of creating a new protocol to transmit data between the private and public networks
- An API to query data from super nodes via http

S.K.N.U Tissera

- Creating a deployment framework to deploy functions and their configuration in a way to suite the decentralized network.
- Security
- Packaging
- Configuration
- Obfuscating functions
- Finding out a mechanism to determine a common access point (an endpoint) to call functions deployed in to the network.
- Finding out a mechanism to route database a scenario where a single node goes done
- Method to retrieve a list of functions to be hosted from super nodes

T.H.A.K Silva

- Finding out a proper mechanism to boot up and scale the decentralized network.
- Decentralized DNS resolution
- Finding out the possibility of using super nodes as DNS servers itself
- Finding an ideal mechanism to cache or persist data in super nodes
- Finding a mechanism to persist data in client application
- Finding out a mechanism to replicate the same function in multiple nodes but the replicated function should act as a single function

A.T.Nimansa

- Payment Model based on requests served by a node, amount of data transferred Via smart contracts
- Finding out the possibility of paying the nodes which hosts functions based on served requests
- Finding out a mechanism to validate the requests served by a mechanism between the origin of the request and the requests served

- Finding out a secure mechanism to prevent DoS attacks which lets a node earns money by sending pings.
- Creating a portal to deploy smart contracts which works as price models
- Finding a method to determine the value of a smart contract.
- Deploying a smart contract to the network with a start price.

DECLARATION

"We declare that the project would involve material prepared by the Group members and that it would not fully or partially incorporate any material prepared by other persons for a fee or free of charge or that it would include material previously submitted by a candidate for a Degree or Diploma in any other University or Institute of Higher Learning and that, to the best of our knowledge and belief, it would not incorporate any material previously published or written by another person in relation to another project except with prior written approval from the supervisor and/or the coordinator of such project and that such unauthorized reproductions will construe offences punishable under the SLIIT Regulations.

We are aware, that if we are found guilty for the above-mentioned offences or any project related plagiarism, the SLIIT has right to suspend the project at any time and or to suspend us from the examination and or from the Institution for minimum period of one year".

	STUDENT NAME	STUDENT NO.	SIGNATURE
1	R.G.D Nayomal (GROUP LEADER)	IT15027948	
2	S.K.N.U Tissera	IT15120212	
3	T.H.A.K Silva	IT15136916	
4	A.T.Nimansa	IT15419910	