“ORB”
DECENTRALIZED FUNCTION AS A SERVICE (DFAAS)

Project ID – 18-070

Preliminary Progress Review Report

T.H.A.K Silva

IT15136916

Bachelor of Science Special (Honors) Degree in Information Technology

Department of Software Engineering

Sri Lanka Institute of Information Technology

Sri Lanka

March 2018

Table of Contents

Table of Figures

1. Introduction

1.1.Purpose

Core Infrastructure of Orb decentralized network is explained here. This will give a basic idea of key components that are existing in the platform, how they interact which each other, how system initially boots up and communication mechanism of the platform network architecture underneath the real architecture and to address the problem of communication gap between public to private network and build bridge between them.

1.1. Scope

Scope of the infrastructure component is to initially boot up the orb platform and bridge the gap between the public and private network. The functionalities of the infrastructure component includes.

I. Mechanism to boot up and scale the decentralized network.

Initially when the system boots up, There are only known super nodes .When user install client application client application know what are super node are existing in network. This nodes are connecting socket streams. Which allow platform to communicate bi direction.

II. Decentralize nodes mapping mechanism which similar to DNS resolution.

When a client get 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.

III. Finding out a ideal mechanism to cache or persist data in super nodes and client.

When a super node or node are connect 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 also its must be persistent and efficient way to storing, retrieve this data. store data can be use to re-initiate connection if failure has occurred.

IV. **Finding out a mechanism to replication the same function in multiple nodes.**

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.

1.3. Overview

The main intention of orb decentralize platform it to provide distribute function hosting platform in order to provide function hosting service platform. The distributed infrastructure is very important, this infrastructure which consist key components of super node and nodes.
which enables the supernode to supernode and node to supernode communication which can be used to achieve node to node communication ultimately.

On top of the infrastructure, the collection of nodes and Super Nodes, a framework is built to let the users consume the services, contribute to the services and to obtain Rewards. Rewards are the key factor to attract the users into the network by allowing the network to scale rapidly.

1.3.1 Main Goals

The main goals of the infrastructure is to provide a efficient and reliable communication layer to the orb platform which enable client application and super node to pass message on real time with minimum latency.

I. **Infrastructure to enable communication between private to public network.**

Orb platform need to communicate between both client and super node which host in private and public networks. Communication between public and private network is possible, but still there is a gap to communicate public to private network. This component main focus is to make it possible.

II. **Provide base level communication API to orb client application which run on PC or a mobile device.**

Orb client application is a software which user can interact to deployed their function to platform and monitor. This application need a communication component and it build on communication API.

III. **Provide base level communication API (Base layer) to super node to connect to both other nodes and clients.**

Orbs super node are hosted in public network , super node are also build on communication API (Base layer) which support to connect to booth super nodes and clients.

IV. **Store critical information which are need to boot up the platform.**

Orb platform need some critical information to initiate its connection and to track them, this are store in application base level communication API.

V. **Enable tunneling between private and public network.**

Once System boot up its possible to communicate bi direction both to public and private network.

VI. **Enable tunneling within same network.**

When client node application are host in private network this can be connect with each other within the network. This feature is support from the base level API.

1.3.3 Major Functionalities and Tasks

The major Task or Functionalities of this component is established connection between all the super nodes and nodes, this can be identified as platform start up mechanism, node connecting mechanism, nodes tracking mechanism and function resolution mechanism.

1.4 Definitions, Acronyms, and Abbreviations

FaaS	**Functions as a Service**

IPFS	Interplanetary File System
DFaaS	Decentralized Functions as a Service
API	Application Programming Interface
Function	A executable code which performs a well-defined specific function

2.0 Statement of work

2.1. Background information and overview of previous work based on literature survey.

There are other distribute platform built for share storage, Crypto currency mining, static content platforms like IPFS, but the specialty of the orb is it provide a decentralized platform to host function as a service,

I. IPFS

IPFS is a distributed file system that seeks to connect all computing devices with the same system of files. In some ways, this is similar to the original aims of the Web, but IPFS is actually more similar to a single Bit torrent swarm exchanging git objects. IPFS could become a new major subsystem of the internet. If built right, it could complement or replace HTTP.

Difference is these systems share/shard static content. But Orb Shares Dynamic content in a decentralized network.

II. Steemit

Steemit is built using blockchain technology and uses a new cryptocurrency to reward users who upload articles, images, commentary, etc. Other ways users can get paid is through sourcing and up-voting popular content. The earlier a person up-votes a post that becomes popular, the more is the reward. Users are paid half in 'Steem Power' (a vesting currency) and half in Steem Dollars, which can be exchanged for US Dollars.

III. Storj

It is a platform, cryptocurrency, and suite of decentralized applications that allows user to store data in a secure and decentralized manner. Storj can be faster, cheaper, and more secure than traditional cloud storage platforms. Faster because multiple machines are serving user his file simultaneously, cheaper because user is renting people's spare hard-drive space instead of paying for a purpose-built data center, and more secure because user’s file is both encrypted and shredded.

2.2. Identification and significance of the problem

IPFS, Steemit, Torrent are examples for distributed computer base concept implementation that we can see in current world. But this system are only allowed to distributed content management

only it work for the static file. But orb platform is provide user to deploy their dynamic content, that can be any function that into distributed concept base platform and earn extra money.

When we are in a same network there are no issues in connecting. But when we think in production scale it faces some issues. We need to tunnel via a server for a proper communication between Public to Private network.

The above-mentioned networks are also built on a decentralized platform. But with Orb user can use dynamic backend services which act as FaaS. If we take it in rewarding system aspect only, the more similar system is Steemit. It has it's own kind of rewards called steem and when users share, comment on their social media platform they get rewarded.

Our system is not having anything related social media but the rewarding system is somewhat like Steemit. Though Steemit uses it for marketing purpose in Orb it acts as the decentralized payment model when it provides Functions as a service

In storj also they act as a system which provides server space but the main dissimilarity is we use it to provide it's uses to host functions and earn out of it. Storj is the data layer for the Internet. Decentralized cloud storage is a new paradigm that removes intermediaries. Storj is open source, distributed, encrypted, and blazing fast object storage. It is more of a storage space and does not have any feature to host a function and give another user an end point to use it like in Orb.

2.3. Technical objectives

We should find a proper mechanism to boot up and scale the decentralized network. This is the task with huge weight. Under this we should find mechanism of initial bootup process, Supernode to supernode communication, node to node communication.

Next objective is to find out a mechanism for DNS resolution in a decentralized network. Initially we thought of having a set of servers as DNS servers but recently we understood that, doing so will reduce the efficiency of the network and make the network less fault tolerant.

Other objective is to find out the possibility of using super nodes as DNS servers itself. It's proposed to have a routing table inside each supernode. This will be broadcasted to the newly joining supernodes.

We should find a mechanism to persist data in the supernodes itself. We are currently storing functions and routing data in memory. But there should persist data when goes to production.

2.4. Detail design

2.4.1 Platform Infrastructure

Platform infrastructure is designing to overcome major problem that existing in internet model that existing in today, in order to communicate between private and public network the backbone is infrastructure that which consist of two key components.

2.4.2 How it's work.

Orb relies on a p2p network. The nodes acts as clients and super nodes act as mediators which helps node discovery. When the number of nodes increases, there must be a sufficient number of super nodes to route requests. Otherwise the network will be flooded. A mechanism to promote node to a Super Node by obtaining a public IP address to a node is under discussion. If the network supports automatic promotion of the nodes to deal with the incoming traffic, the resiliency of the network will increase in vast amount

2.6. Anticipated benefits

This component brings an ground layer communication platform for Orb network to work.

I. Base layer to orb platform to build
II. Enable communicate public to private network
III. Work similar to DNS resolution.
IV. Tunneling Public to Public
V. Tunneling public to private.

3. Project plan or schedule

Project Start Date 4/1/2018 (Sunday) Display Week 20
Project Lead

WBS	TASK	START	END	DAYS	WORK DAYS
1.16	Mechanism to boot up and scale the decentralized network	5/13/2018	7/5/2018	18	39
1.17	Decentralized DNS resolution	5/13/2018	7/19/2018	19	49
1.18	Possibility of using super nodes as DNS servers itself	6/8/2018	8/14/2018	20	48
1.19	Mechanism to cache or persist data in super nodes	8/10/2018	8/14/2018	21	3
1..20	Mechanism to persist data in client application	Sun 5/13/18	Thu 7/05/18		
1.21	Mechanism to replicate the same function in multiple nodes but the replicated function should act as a single function	Fri 6/08/18	Tue 8/14/18		

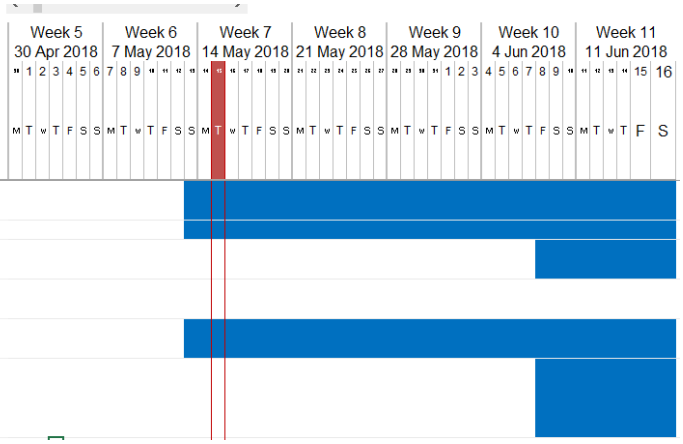

Figure 1 : Gantt Chart Part 1

Project Start Date 4/1/2018 (Sunday) Display Week 20
Project Lead

WBS	TASK	START	END	DAYS	WORK DAYS
1.16	Mechanism to boot up and scale the decentralized network	5/13/2018	7/5/2018	18	39
1.17	Decentralized DNS resolution	5/13/2018	7/19/2018	19	49
1.18	Possibility of using super nodes as DNS servers itself	6/8/2018	8/14/2018	20	48
1.19	Mechanism to cache or persist data in super nodes	8/10/2018	8/14/2018	21	3
1..20	Mechanism to persist data in client application	Sun 5/13/18	Thu 7/05/18		
1.21	Mechanism to replicate the same function in multiple nodes but the replicated function should act as a single function	Fri 6/08/18	Tue 8/14/18		

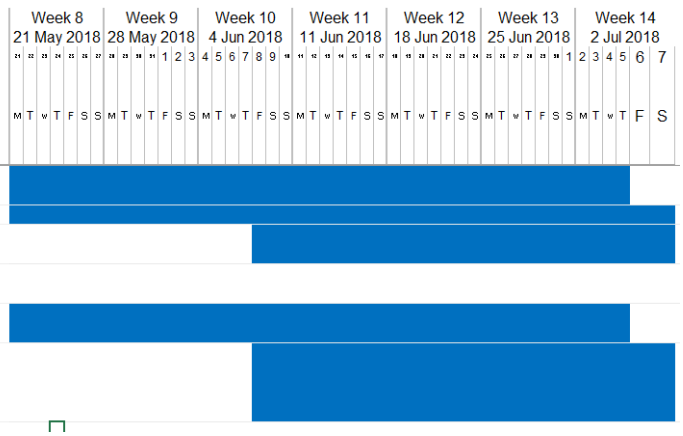

Figure 2 : Gantt Chart Part 2

Project Start Date 4/1/2018 (Sunday) Display Week 20
Project Lead

WBS	TASK	START	END	DAYS	WORK DAYS
1.16	Mechanism to boot up and scale the decentralized network	5/13/2018	7/5/2018	18	39
1.17	Decentralized DNS resolution	5/13/2018	7/19/2018	19	49
1.18	Possibility of using super nodes as DNS servers itself	6/8/2018	8/14/2018	20	48
1.19	Mechanism to cache or persist data in super nodes	8/10/2018	8/14/2018	21	3
1..20	Mechanism to persist data in client application	Sun 5/13/18	Thu 7/05/18		
1.21	Mechanism to replicate the same function in multiple nodes but the replicated function should act as a single function	Fri 6/08/18	Tue 8/14/18		

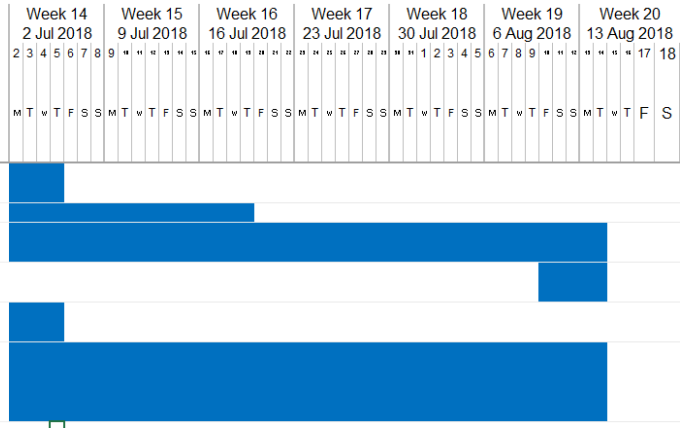

Figure 3 : Gantt Chart Par 3

Task Number	Task	Effort (weeks)
1	Finding out a proper mechanism to boot up and scale the decentralized network.	4
2	Decentralized DNS resolution	3
3	Finding out the possibility of using super nodes as DNS servers itself	2
4	Finding an ideal mechanism to cache or persist data in super nodes	5
5	Finding a mechanism to persist data in client application	4
6	Finding out a mechanism to replicate the same function in multiple nodes but the replicated function should act as a single function	3

4. Research constraints

When we are analyzing ways to do Decentralized DNS resolution, we thought of having a set of servers as DNS servers but recently we understood that, doing so will reduce the efficiency of the network and make the network less fault tolerant.

So currently we are storing data and routing data in memory. But at production stages it's hard to do so. So, we should keep a persistence database. So that also can be taken as a constraint in this regard.

There are technologies like Couchbase which is a distributed database, we are in the process of finding a database which can synchronize with the known peers.

If a super nodes restarts, there is no way of retrieving data if not persisted.

5. Specified deliverables

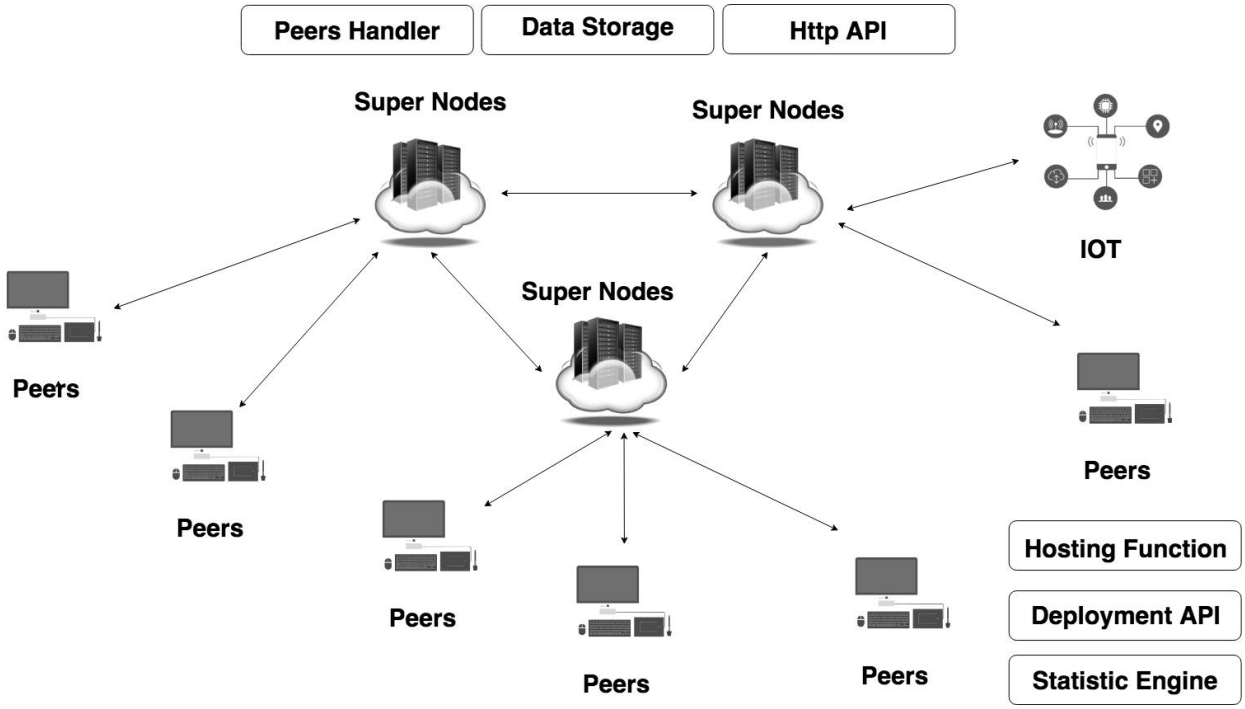

Figure 4 : infastructure

This give a great idea about the specified deliverables in the system. All these entities should work in common platform for communication. That's what it's focused in this deliverable.

We should also find a way to route request in the system.

6. References

[1]"What is obfuscation (obfu)? - Definition from WhatIs.com", SearchSoftwareQuality, 2018. [Online]. Available: https://searchsoftwarequality.techtarget.com/definition/obfuscation. [Accessed: 07- May- 2018]

[2]"An Introduction to IPFS – ConsenSys – Medium", Medium, 2018. [Online]. Available: https://medium.com/@ConsenSys/an-introduction-to-ipfs-9bba4860abd0. [Accessed: 07- May- 2018]

[3]"What is a Torrent? - Definition from Techopedia", Techopedia.com, 2018. [Online]. Available: https://www.techopedia.com/definition/5263/torrent. [Accessed: 07- May- 2018]
[4]"How Do Ethereum Smart Contracts Work? - CoinDesk", *CoinDesk*, 2018. [Online]. Available: https://www.coindesk.com/information/ethereum-smart-contracts-work/. [Accessed: 14- Jan- 2018].

[5]"ConsenSys – Medium", *Medium.com*, 2018. [Online]. Available: https://medium.com/@ConsenSys. [Accessed: 21- Jan- 2018].

[6]"What Is AWS Lambda? - AWS Lambda", *Docs.aws.amazon.com*, 2018. [Online]. Available: https://docs.aws.amazon.com/lambda/latest/dg/welcome.html. [Accessed: 22- Jan- 2018].

[7] "Steemit," *Steemit*. [Online]. Available: https://steemit.com/. [Accessed: 24-Mar-2018].

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

[10]"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].

[11]"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].

[12]"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].

[13]P. Labs, "IPFS is the Distributed Web", *IPFS*, 2018. [Online]. Available: https://ipfs.io/. [Accessed: 14- Jan- 2018].

[14]"Storj - Decentralized Cloud Storage", *Storj - Decentralized Cloud Storage*, 2018. [Online]. Available: https://storj.io. [Accessed: 14- Jan- 2018].

[15]*Google Cloud Platform Documentation | Documentation | Google Cloud*. [Online]. Available: https://cloud.google.com/docs/. [Accessed: 02-Feb-2018].

[16]"Kubernetes Documentation," *Kubernetes*. [Online]. Available: https://kubernetes.io/docs/home/?path=users&persona=app-developer&level=foundational. [Accessed: 02-Feb-2018].

[17]"Decentralized Internet on Blockchain – Hacker Noon," *Hacker Noon*, 03-Oct-2017. [Online]. Available: https://hackernoon.com/decentralized-internet-on-blockchain-6b78684358a. [Accessed: 02-Feb-2018].

[18]*Blockchain Based Distributed Control System for Edge Computing - IEEE Conference Publication*. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/7968630. [Accessed: 02-Apr-2018].

[19]"Decentralizing Your Microservices Organization," *The New Stack*, 31-Oct-2017. [Online]. Available: https://thenewstack.io/decentralizing-microservices-organization/. [Accessed: 02-Apr-2018].

[20]"Building Your First Network," *Building Your First Network - hyperledger-fabricdocs master documentation*. [Online]. Available: http://hyperledger-fabric.readthedocs.io/en/release-1.0/build_network.html. [Accessed: 02-Apr-2018].

[21]M. Meister, "leveraging Kubernetes to run a private production ready Ethereum network," *Medium*, 26-Nov-2017. [Online]. Available: https://medium.com/@cryptoctl/leveraging-kubernetes-to-run-a-private-production-ready-ethereum-network-b6f9b49098df. [Accessed: 02-Apr-2018].

[22]*A Decentralized Service Discovery Approach on Peer-to-Peer Networks - IEEE Journals & Magazine*. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/5928313/. [Accessed: 02-Apr-2018].

[23]"Taming WebRTC with PeerJS: Making a Simple P2P Web Game," *Toptal Engineering Blog*. [Online]. Available: https://www.toptal.com/webrtc/taming-webrtc-with-peerjs. [Accessed: 02-Apr-2018].

[24] "Decentralized payments for environmental services: The cases of Pimampiro and PROFAFOR in Ecuador," Ecological Economics, 31-Dec-2007. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921800907005320. [Accessed: 05-Jan-2018].

[25] A. Crespo and H. Garcia-Molina, “Semantic Overlay Networks for P2P Systems,” *SpringerLink*, 19-Jul-2004. [Online]. Available: https://link.springer.com/chapter/10.1007/11574781_1. [Accessed: 02-Mar-2018].

[26] “Rethink the Web browser,” Beaker | Peer-to-peer Web browser. No blockchain required. [Online]. Available: https://beakerbrowser.com/. [Accessed: 05-Feb-2018].

7 . Appendix

Appendix A

```
var express = require('express'),//express js
    bodyParser = require('body-parser'),//initial body parser
    port = process.env.PORT || 3300,// port assign
    app = express(),//express initiate
    server = require('http').Server(app),
    io = require('socket.io')(server),
    p2p = require('socket.io-p2p-server').Server;

    io.use(p2p);

    var nodes=[];
    var sock =null;

io.on('connection',function (socket) {

    sock = socket;
    console.log('Stream Build \n ID : '+socket.id);

    var arr={count:'0',socket:socket.id,port:'3100'};

    nodes.push(arr);

    socket.on('peer',function (data) {

        socket.emit('monitor', { data:{nodes:nodes}});
        console.log("peer namespace call");
    });

    socket.on('get-function',function (data) {
        console.log('server Call');
        console.log(data.length);
        if(data){
          socket.emit('returnfunction', "this is cool , function request by \n peer : "+data);
        }else{
            console.log('empty result');
        }
    });

    socket.on('disconnect',function (data) {

        console.log('close: '+ socket.id);

        //todo: socket info must remove from table

    });
```

```
    socket.broadcast.emit('returnfunction', "this is cool , fu");

    socket.emit('monitor', { data:{nodes:nodes}});

});



app.use(bodyParser.urlencoded({ extended: true }));
app.use(bodyParser.json());

app.use('/public', express.static(__dirname +'/public'));


app.get('/',function(req,res){
    res.sendFile(__dirname +'/public/index.html');
});


server.listen(port,"0.0.0.0",function(){
    console.log("Externals:"+port);
});
```