iagon

Intelligent and Decentralized Marketplace for Distributed Computing

Architecture and Technical Paper

June 15th 2018

Outline

Motivation

In this paper we provide a detailed overview of our vision of building an AI driven marketplace for completely decentralizing cloud computing, including the details of the prototype we've built and what's coming next. There are plethora of cloud computing solutions in the market today. These include the ones from the Big 4 providers that are Amazon [1], Microsoft [2], Google [3], IBM [4] and also several newcomers like Golem [5], Sonm [6], IExec [7], Boinc [8], Droplet [9], etc. that build cloud computing solutions based on utilizing idle compute capacity in the network. One important limitation with all these solutions however is that they have some centralized component that is critical for the overall functioning of the system. Our goal is to further stretch the boundaries in distributed computing and build a completely open, decentralized, infinitely scalable cloud computing solution that use flexible and secure marketplaces for trading compute resources. The idea of the decentralized marketplace is similar to what is proposed in [10].

The advent of the Blockchain, Distributed Ledger Technology (DLT) and cryptocurrency technologies have also made it possible to build Internet scale solutions comprising of non-trusting computers to be audited and incentivized for their work. Both Blockchain and peer-to-peer (P2P) networks have opened doors for building truly decentralized Internet scale cloud computing solutions with almost infinite resources. These trends and technologies are therefore being leveraged for building a decentralized marketplace and ecosystem for trading compute resources and building Internet scale cloud computing solutions in an very unique way that has not been done before. More specifically, any job that can be encapsulated as a Docker [11] container can be submitted to this new Iagon cloud computing platform, and clients only have to pay for the duration of the execution of the job. Users will be able to tap into almost unlimited highly available compute capacity at fraction of a cost for hosting their Docker containers. These marketplace computing services are offered in tiers, which specifies the levels of computing performance and trustrability.

Iagon's solution offers a simple concept, yet it is very difficult to implement. This is because it needs to add and coordinate multiple technologies and optimize for best performance. Iagon's decentralized computing services not only will offer individual developers and small companies to tap into large on-demand compute resources for low prices, but also will allow larger companies to leverage grid cloud computing without handing over the reins of their critical computing operations to another company. For example, Netflix running its entire productions operations in AWS which is owned by one of their biggest competitor.

In the remainder of this paper it will be provided the details of Iagon system, the current development status, the threat model and future roadmap.

Iagon Architecture

Iagon system architecture is shown in Figure 1 below, connectting client interface (Windows, Mac OS and Linux) to the distributed computing resources network infrastructure through an intelligent and decentralized marketplace, using Blockchain as a control plane layer that authenticates, validates and secures the computational resources using an Artificial Intelligence layer that allocates, schedules and optimizes the distributed computing resources and matches with the cliet job requests. This layer is called the “Alexandria Protocol”.

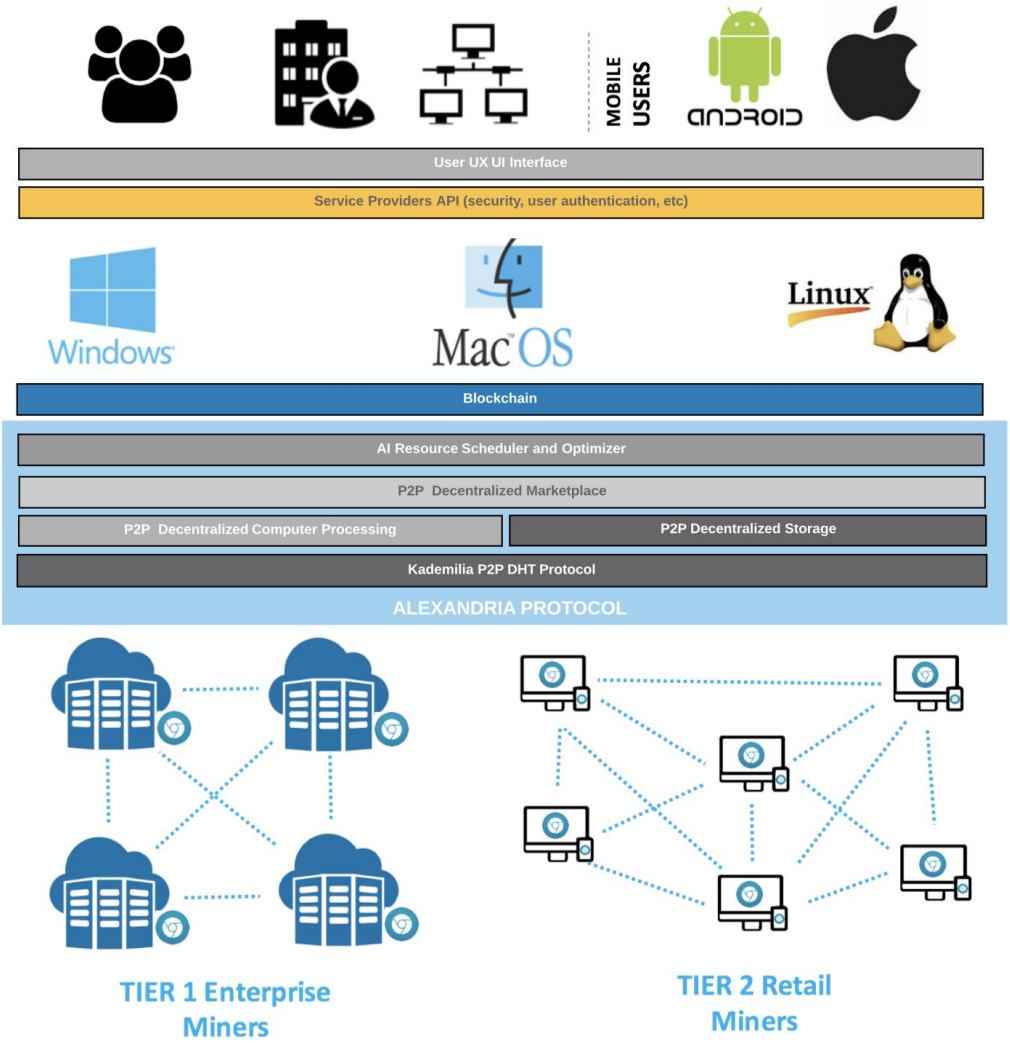

Figure 1 - Iagon’s System Architecture overview.

The design of the decentralized P2P Blockchain marketplace for distributed computing resources (storage & computation) is shown in Figure 2. This architecture is formed by three layers

i) DApps- (Decentralized Applications) and Web-client interface layer;

ii) Blockchain Ethereum Layer, and the

iii) P2P Network Layer.

The DApps/Web-client layer is where users run their computational jobs or requests to the decentralized computing resource infrastructure, composed of Blockchain (control, consensus and marketplace layer) and P2P network layers. The Blockchain layer is where all computing resources are segmented and published in service tiers in a marketplace. These services are allocated, using encrypted hash tag pointers that relates to a particular computing node in the network, where the resources are alocated or processed. The P2P layer is comprised of several netwok nodes, called “Utilitarian”, connected via a Kademilia P2P network technology, having a three function as:

i) Resource Computational Node: it processes and stores files and programs, providing distributed computational resources to the marketplace, and as

ii) User Client Interface: it work as a client interface that requests computing services to the P2P network.

ii) Marketplace or resource exchanges: it also works as decentralized P2P computing resource sharing and commercialization.

The underlying P2P platform is based on the Kademilia network where any new nodes can join the Kademilia, be inserted and synchonized to other peers. Once these nodes are added, the computing resource users (utilitarians) can then configure the way they want their computing resources to be available in the marketplace and their selected rewards for these services. The services are then added to the Blockchain, categorized by different tiers of services, where the nodes can automatically sign up to publish their resources availability using encrypted hash pointers. Each tier creates a unique set of computing resources where clients and computing node providers can interact to commercialize these assets in a decentralized manner. Once the service match is identified then a smart contract transaction takes place, exchanging Iagon’s tokens (IAG) to Utilitarian nodes to pay for the services rendered to the system.

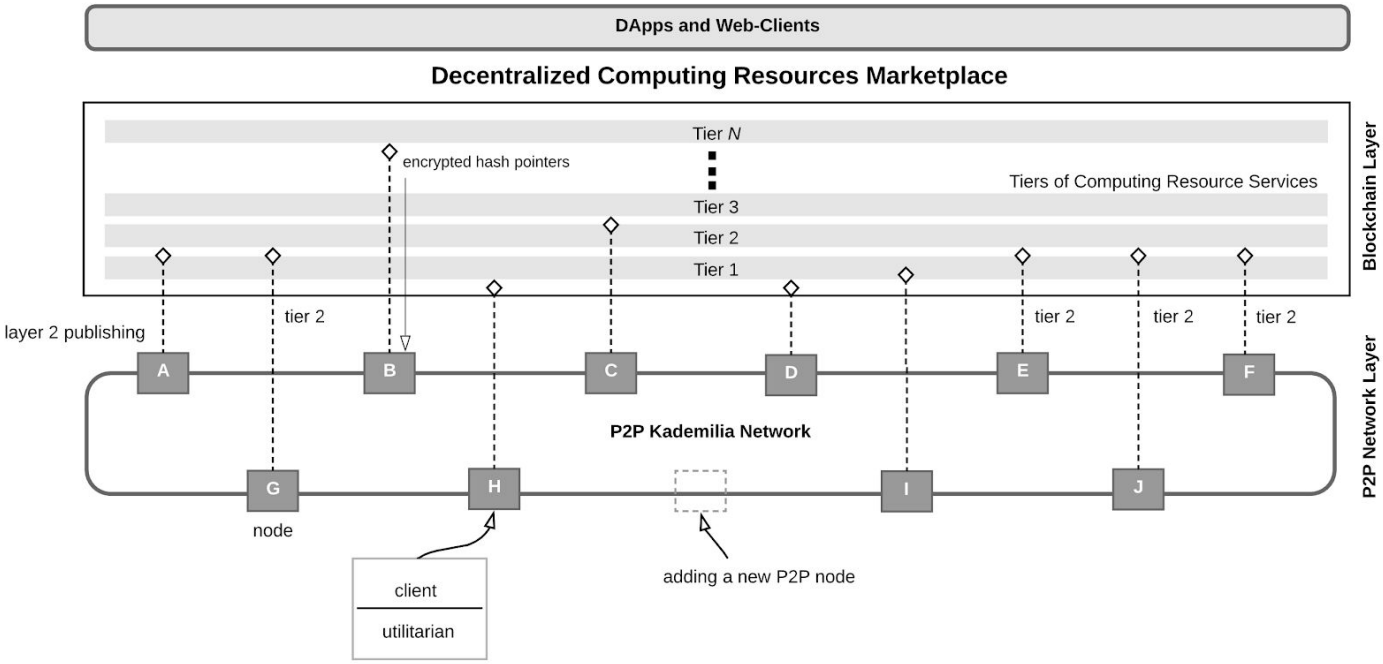

Figure 2 - Iagon's P2P Decentralized Marketpace for Distributed Computing Resource.

Distributed Computing Service Tiers

As shown in Figure 3, Iagon's decentralized computing system comprises of P2P tiers, where computing resources are share using an Ethereum Blockchain for managing transactions involving compute resources.

In summary, the main elements of this network technology are:

1. **Clients**: Nodes that are looking for compute resources for executing their tasks and are willing to make payment for those resources;
2. **Utilitarian**: Nodes that want to sell their spare compute resources for a reward;
3. **Marketplace owners or exchanges**: Dynamically selected nodes that facilitate clients to discover utilitarians. There can be multiple marketplace owners in the network depending upon the range of compute resources that utilitarians sell.

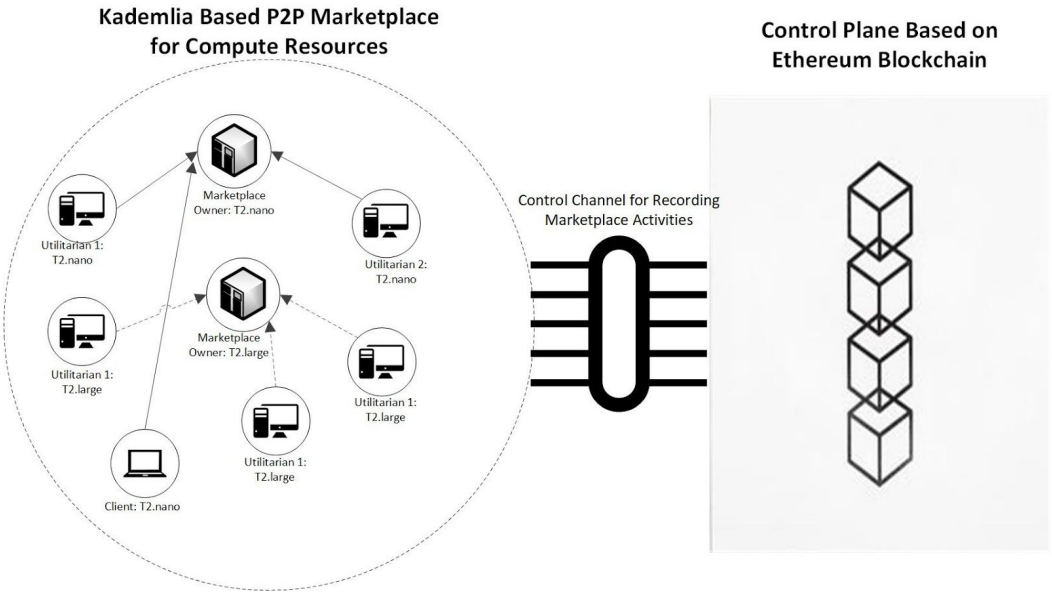

Figure 3 - Iagon's Tiers of Decentralized Computational Resources.

As shown in Figure 3, there are two marketplaces for compute resources T2.nano and T2.large in the network. The utilitarians looking to sell these compute resources list themselves in one of these marketplaces. Similarly, the client for T2.nano compute resource lookups the corresponding marketplace owner and gets a list of utilitarians who are able to provide that service.

All the nodes in Iagon system have public Ethereum addresses. It is assumed that nodes are rational entities that participate in order to maximize the value they can generate from the network, where game theory principles apply. On the other hand, there can also be some malicious nodes in the network and a discussion on the ways to minimize their impact on network operations will follow next.

Broadly speaking here are the steps involved in Iagon's system operations:

1. Nodes join the Kademlia P2P network. The Kademlia id is generated as follows:
    a. Kademlia id = hash(Ethereum public address, IP address, country code)
    b. Iagon have a list of published nodes that help new nodes to join the network. But going forward Iagon will have an API that returns a list of nodes from the Blockchain that returns a random list of verified nodes that are already part of the P2P network.
2. Worker (aka utilitarian) nodes decide the tier of service they can provide. This is based on the number of CPUs and RAM. The utilitarian nodes need to have a CPU utilization of under 50% in last 1 hour to be eligible to sell computing resources. There is a tier definition for different classes of compute resources.
3. Utilitarian nodes generate a Kademlia id for the tier of service they can provide and do a lookup corresponding to that Kademlia id. The returned node is the marketplace owner for that tier. However, in order to make sure that one single marketplace owner doesn't monopolize a given tier of service, Iagon add the week number in the hash function as well.
    a. Kademlia id of Marketplace owner = hash(vCPUs, RAM, Week number)
4. Utilitarian nodes register themselves with the marketplace node. The registration information is a tuple of the form (IP address, time interval of availability). The copy of this registration is also stored in Blockchain for auditing purposes.
5. Users of the client nodes can specify the tier of the service they require for their task that is specified as a Docker image. Based on the tier selected the client nodes lookup the corresponding marketplace owner. The lookup process here is same as what the utilitarian nodes used above in Step 3.
6. The client node gets a list of all the utilitarian nodes from the marketplace owner.
7. The client node then does an auction. During this auction it contacts all the utilitarian nodes for their pricing information. Iagon uses a granularity of 15 mins for specifying the price of execution. This step also serves as a verification that the worker nodes are still able to share their computing resources. Also, the client node can measure the latencies to all of the utilitarian nodes. All the utilitarian nodes that have a latency more than some client specified threshold (default 5 secs) are rejected.
8. Upon receiving the pricing bids from the utilitarian nodes, the client node selects the node with the lowest bid. However, the price that is paid to the winner is the second lowest price. This is called Vickrey auction. This form of auction mechanism ensures that workers best bidding strategy is to truthfully share their cost of providing computing resources. Auction details are also recorded in Blockchain.
9. The client node communicates with the utilitarian node and sends the Docker image to be executed. The results of the computations are sent back to the client and stored in a predetermined directory or through a callback URI.
10. The marketplace node is also paid by the client. The amount of this payment is the difference in amount between the lowest and second lowest bid.
11. The steps above, including the auction process, task assignment and completion, and final payments will all be governed by a smart contract.

Current Development Status

Iagon have built a prototype for the new distributed computing solution using the marketplace concept as described above. The implementation is part of the Iagon's ElectronJS application that users install. The application allow users to join the Kademlia based P2P network, allow utilitarians to sell their spare compute cycles, and also allow clients to discover utilitarians for executing tasks encapsulated as docker images. Below it is explained how Iagon defines different tiers of compute resources and how utilitarian and clients configurations work.

Service Tier Definition

It is assumed that utilitarians in the network are predominantly home users with laptop and desktop machines. This is defined as Tier-2 utilitarians. In the future, Iagon will have enterprise grade hardware, software providers and datacenter operators to also join the Iagon platform to sell compute power. These are defined as Tier-1 utilitarians. The Tier-2 level is further subdivided into several sub-categories that represents a range of computing power as shown below. For example, T2.small represent any machine with upto two CPUs, between 2 and 4 GB of RAM, and with the CPU speed of up to 2 GHz. The tiering and sub-categorization strategy accounts for future addition of Tier-1 providers. This service tiers are listed in Table 1 below.

Table 1 - Iagon's Service Tiers Categorization based on Utilitarian Computing Resources.

Tier Level	OS	Upto Number of CPUs	Upto Memory (RAM in GB)	Upto Speed (GHz)	Instance Name
2	Windows/Linux	2	2	2	T2.nano
2	Windows/Linux	2	4	2	T2.small
2	Windows/Linux	2	8	2	T2.medium
2	Windows/Linux	2	16	2	T2.large
2	Windows/Linux	2	32	2	T2.xlarge
2	Windows/Linux	2	2	4	T2.nano.fast
2	Windows/Linux	2	4	4	T2.small.fast
2	Windows/Linux	2	8	4	T2.medium.fast
2	Windows/Linux	2	16	4	T2.large.fast
2	Windows/Linux	2	32	4	T2.xlarge.fast
1	Windows/Linux	More than 2			T1.default

Utilitarian Configurations

The Iagon agent determines the number of CPUs and RAM for the node and automatically determines the tier the nodes' resources fall into. The agent then also looks up the marketplace owner for that tier and list the node with the marketplace owner for selling the compute resources. The users have the option to list the time period during which the compute resources should not be used by others. Also, the user can provide the price in USD for every every hour of sharing their compute resources. The clients however are charged in the increments of 15 mins for using utilitarian resources. Once a node is listed at a marketplace for providing compute services then it is referred to as a utilitarian.

As shown below in Figure 4, the processing settings section in the Iagon app allow utilitarians to configure values for selling their compute capacity.

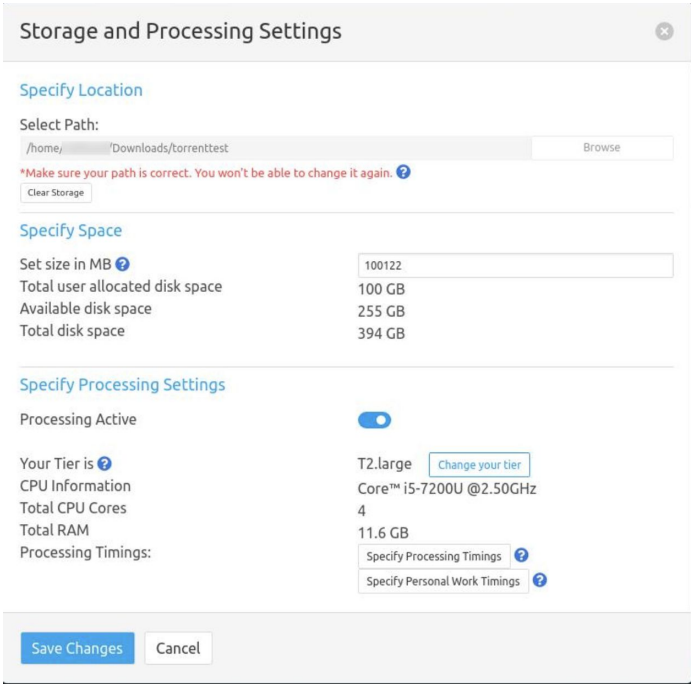

Figure 4 - Iagon's Utilitarian Computational Resources configuration.

Client Configurations

The users of the Iagon app can also choose to buy compute resources from others in the network for distributed execution of their tasks. These nodes or users are referred to as clients.

Figure 5 of the Iagon app shows how the clients are able to configure their requirements for the tier of compute capacity that they require for executing their tasks.

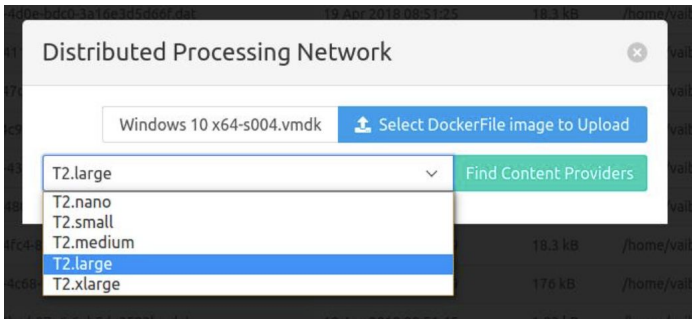

Figure 5 - Iagon's Client Interface.

After the client node has specified it's task requirements and specified the Docker image to be executed, the Iagon agent reaches out the marketplace owner for that tier and get a list of available utilitarians. The user is provided the details of the utilitarian along with the option for selecting one as shown below. This is shown in Figure 6 below.

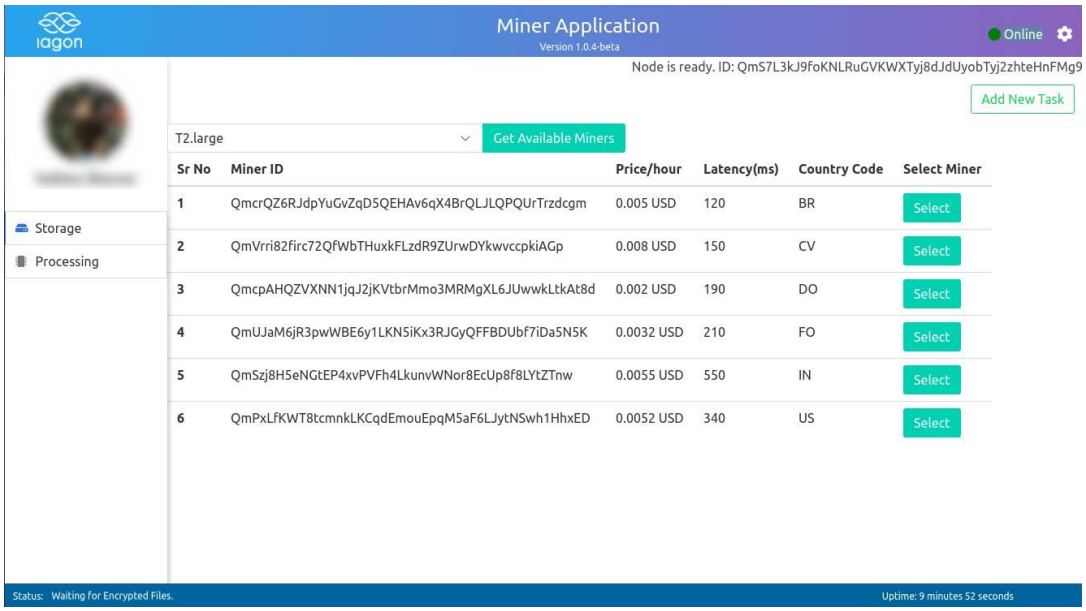

Figure 6 - Iagon's list of Utilitarian Resources provided to the Marketplace.

If a user does not select one of these resources, then the agent can also be configured to automatically select the utilitarian with the lowest possible cost as long as the latency to the utilitarian from the latency test is under 2 secs. The client node can also configure a directory where the test results are stored. Once the results from the computation are available the user has the option to receive an email confirming the work done. Also, the Iagon app provides a notification for the same actions.

Threat Model Scenarios and Solutions

Since Iagon is building a completely decentralized system with no centralized governance there are potentially various scenarios in which different participants might try to game the system. This section describes several such key scenarios and propose Iagon's solution for ensuring that the overall system continue to function with high performance and fidelity. The solutions proposed here are a mix of innovative technical protocols as well as clever incentive engineering, which help to ensure that abiding by system rules is the most rational strategy for the participants.

Eclipse Attack

In the Eclipse attack [12] an adversary can eclipse an individual node from participating in a P2P network. Such an attack is possible if say more than 50% of network nodes are controlled by an adversary. In [13] the authors have recommended that by adding IP address along with the Ethereum public address can help to mitigate the impact of Eclipse attack. In Iagon implementation, this recommendation has been used to generate the Kademlia id for nodes, such as:

Kademlia id = hash(Ethereum public address, IP address, country code)

So the above strategy should help mitigating the risks associated with an Eclipse attack.

Sybil Attack

The Sybil attack [14] is an extended version of the Eclipse attack wherein an adversary is able to control most of the nodes in the network so as to bring down the overall reputation and functioning of the network. In fact this attack is a prerequisite for the Eclipse attack to succeed. One manifestation of the Sybil attack in Iagon's system is that an attacker can control the marketplace and utilitarians and take control of client computations wherein they get paid for their work without doing any actual work. A client who is relying on a single utilitarian or set of utilitarians for performing the work for them will have no way to know whether the output received is correct or fake. So it is an important variation of the Sybil attack that can happen in the system.

The mitigation strategy described above for the Eclipse attack can be useful. There are couple of other techniques that can be employed that will also help for "good" nodes in the network to be able to minimize the impact of a Sybil attack. These techniques revolve around reputation management and cross-checking computation results. These techniques are explained in detail in the future work section.

Greedy Utilitarians

In this attack it is possible for utilitarians to submit a low cost bid for tasks but then provide a poor quality of service for clients. Since the clients will not know immediately that utilitarians provided poor quality or incorrect computation on the tasks provided to them. This is a form of Sybil attack, but on a small scale wherein there are greedy utilitarians who want to get compensation for tasks without actually completing those tasks. The techniques proposed for dealing with the Sybil attack will also be useful for both avoiding these utilitarians from winning the auction process and also from detecting output wherein utilitarians did not perform the necessary computation.

Malicious Marketplace Owners

In this attack scenario it is considered the impact of having malicious marketplace owners in the network. Here are the kinds of attacks that are possible - a) colluding with the malicious utilitarians and suppressing good nodes from participating in the auction process, b) not storing and/or sharing utilitarians' information with the clients in an effort to diminish overall system utility. Iagon addresses these problems in the following manner as part of the future development of the solution:

1) Building a reputation for the marketplace owners similar to the way of building reputation for the utilitarians (described more in the future work section).
2) Rotating the marketplace owners every week for a given tier of service. As explained in the system overview section for computing the hash of a tier one of the input values is the week number of the year. So every week the utilitarians even for the same tier re-list themselves with a new marketplace owner. The clients are able to find the new marketplace owners since they also keep updating the hash they use to do a lookup for them. It should be noted that all times in Iagon system are based on UTC. Also, it is not required globally synchronized clocks. If a client does a lookup for the marketplace owner for a tier and no utilitarian information is received, the system automatically will retry for the new marketplace owner by bumping up the week number by 1.
3) As part of Iagon's future work, there will be redundant marketplace owners for every tier. The redundant nodes will be the immediate successor neighbors of the designated marketplace owner. So for example, say Node 1 is the marketplace owner for Tier-1 then utilitarians will also list themselves in the immediate successor which is Node 2. The clients when getting the list of utilitarians from Node 1 will also contact Node 2 and get the list of utilitarians. If the two sets of data vary significantly even after contacting the utilitarians then the client can skip the payment to Node 1 and also broadcast the poor reputation for the node.

Free-loading Clients

It's possible that clients can also mis-use the resources in the network by say getting their tasks executed, but not marking the payments to the utilitarians and marketplace owners. This is solved by using Blockchain as an escrow and enforcing the transaction through a smart contract.

Future Work

While Iagon made great headway in building a marketplace for distributed computing there some important innovative features that are currently working that willl make Iagon's system truly best in class for a wide variety of workloads while at the same time providing lowest possible costs and highest reliability to Iagon clients. Below is a glimpse of the most important categories of those feature sets being developed.

Public REST APIs

Currently the marketplace offering is integrated into Iagon's existing app that also allow users to sell and buy storage capacity. Iagon is also working on exposing the same capability through RESTful APIs for selling, buying and managing compute resources. It is expected that this open platform will allow developers to build new innovative apps to leverage massive, inexpensive and easy to access compute resources. This is shown in Figure 7.

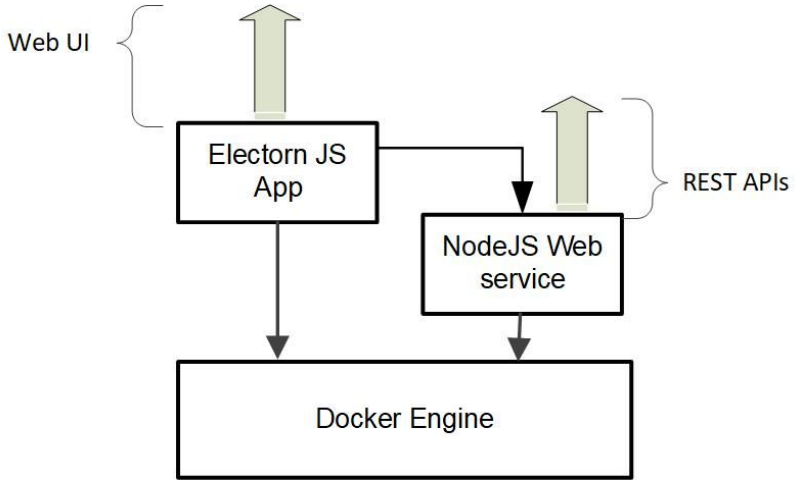

Figure 7 - Iagon's Client Configuration and Open Developer's Interface Using Rest APIs.

Here's a summary of how these APIs being developed looks like:

Create a compute resource to sell
POST /compute
{
"tier_name": "string",
"kademlia_id": "string",
"public_address": "string",
"ip_address": "string",
"country_code": "string",
"price_per_15_mins": "double",
"availability_window": "string",
"cpu_count": "int",
"speed_in_ghz": "int",
"memory_in_gb": "int"
}

Get a list of utilitarians providing a particular tier of compute resource
GET /compute/{tier_name}
[
{"kademlia_id": "string",
"public_address": "string",
"ip_address": "string",
"country_code": "string",
"latency_in_msecs": "int",
"price_per_15_mins": "double",
"cpu_count": "int",
"speed_in_ghz": "int",
"memory_in_gb": "int"},
{},...
]

Submit a Docker instance for execution on the selected utilitarian
POST /compute
{
"client_kademlia_id": "string",
"client_public_address": "string",
"client_ip_address": "string",
"docker_image": "blob",
"return_uri": "string"
}

Workflow Management

The current implementation manages a task for a client that is packaged as a single docker container. There are variety of workloads that require a set of interdependent tasks that need to be executed sequentially with some intermediate parallel steps. So Iagon is building a general workflow management system that clients can use to define and submit a workflow of tasks. In turn the workflow management system will automatically schedule, manage and optimize the execution of all the tasks so as to provide best reliability, performance and cost benefits for completing all the necessary tasks.

Region Demarcated Compute Resources

Currently Iagon is creating a single P2P network and clients have the option to select utilitarians from a specific country and/or utilitarians with certain latency characteristics. Iagon is also working, to support multiple P2P networks that are region specific. For example, a P2P network for US West, US East, EU West, EU East, India, South-East Asia, etc. This not only will simplify selecting utilitarians that are geographically close, but also make it possible to meet region specific data handling requirements like the GDPR regulations in European Union.

Reputation Management

Since Iagon has no central authority in the decentralized computing resource system, the use of a clever reputation management and incentives engineering is used to enable the system to be self-sustainable. Iagon wants to make sure that malicious or non-performing utilitarians and marketplace owners, as described in the previous sessions, will get weeded out from the system, and at the same time freeloading from clients is avoided. Iagon's reputation management is inspired by the work of trust groups as described in [15]. Every node in the network has a copy of everyone else's reputation. This reputation is an aggregate representation of the node's direct experience with working with the other nodes, and also the reputation broadcast messages that the node has received. This reputation is calculated for every other node be it a utilitarian, marketplace owner and client.

Basically this is how it works by taking the example for evaluating utilitarians:

1. The reputation is associated with the Kademlia id of a node, which in turn means that it's associated with the Ethereum public address.
2. Reputation is a monotonically increasing integer. Higher the value means higher the reputation and 0 being the worst. The value of 0 also means that a node's reputation is unknown. Iagon treats worst reputation and unknown interchangeably since a malicious node can always regenerate it's Kademlia id and re-join the network as an unknown node.
3. A utilitarian after successfully completing a transaction creates a completion certificate and broadcasts to all nodes it's aware of in the network. The completion certificate basically contains a hash pointer to the Ethereum block that records the payment transaction from the client to the utilitarian. A node after receiving the completion certificate calculates the reputation of the utilitarian as follows:
    3.1. utilitarian reputation new = f(utilitarian reputation old * client reputation) or 1 if either of the two values are 0
    3.2. For the same pair of utilitarian and client nodes increase the reputation at most once in a week
4. Reputation of a node is decaying function of time. So if a utilitarian does not provide service it gradually degrades over time.
    4.1. New Reputation = Ratings in last 30 days * α + Previous ratings * (1 - α), where α controls the weightage assigned to newer ratings

Reputation of nodes will be an integral part of the determination when nodes decide to offer and/or receive services.

Trojan Injection for Results Verification

One of the results verification technique that is employed to make sure utilitarians are not just returning junk results back to clients is called Trojan injection. In this technique automatically inject a step in client computation that has a known output value. When a task is completed the output results set should have this known value included in the output results set. If it is missing then it will be known that a utilitarian has not processed the task as per the client's specification and therefore should not be paid. This technique is similar in principle to the Proof-of-Work concept (PoW) used in Bitcoin network with the goal to ensure that the utilitarian is indeed expending it's computing power.

Machine Learning/AI

Underlying the entire computing platform is a machine learning algorithm that continuously learns from the interactions in the network and optimizes the strategy for every participant in the network, it is described as:

1. Builds a reputation for other nodes in the system, be it utilitarians, clients or marketplace owners;
2. Predicts the uptime and availability of utilitarians;

3. Predicts the approximate completion time of tasks for the clients based on task specifications;
4. Recommends the best pricing strategy for the utilitarians so as to maximize local resource utilization as well as profit potential.

As Iagon compute platform scales with more tasks and participants, the machine learning models will learn from the additional data and will increasingly become more useful for the participants. This area is ripe for further innovation and Iagon plan to build open APIs for external developers and research community to also contribute newer machine learning models that individual nodes can pick from and utilize.

Summary

Iagon is building the best-in-class, innovative, high performance and cost-effective Distributed Computing Solutions, based on an unique combination of advanced technologies such as P2P, Blockchain and Machine Learning/AI. The underlying principle of all this development is to offer users/clients the best quality and affordable computing services they can have and offer the utilitarian workers the best reward mechanism to incentive them to offer and utilize their spare computational resources.

The Iagon's CTO's office development team is working hard to execute on these deliverables and solve the challenges ahead to delivering unique and innovative solutions to the market.

References

[1] https://aws.amazon.com/
[2] https://azure.microsoft.com
[3] https://cloud.google.com
[4] https://www.ibm.com/cloud/
[5] https://golem.network/
[6] https://sonm.com/
[7] https://iex.ec/
[8] https://boinc.berkeley.edu/
[9] https://www.digitalocean.com/products/droplets/
[10] R. Gupta, and A. K. Somani. CompuP2P: An Architecture for Internet Computing Using Peer-to-Peer Networks. In Proceedings of the IEEE Transactions on Parallel and Distributed Systems (Volume: 17, Issue: 11, Nov. 2006).
[11] https://www.docker.com/
[12] https://www.usenix.org/node/190891
[13] https://www.cs.bu.edu/~goldbe/projects/eclipseEth.pdf
[14] https://en.wikipedia.org/wiki/Sybil_attack
[15] R. Gupta, and A. K. Somani. A Framework for Reputation Management and Using Reputation as Currency in Large-Scale Peer-to-Peer Networks. In Proceedings of the Fourth IEEE International Conference on Peer-to-Peer Computing, ETH Zurich, Switzerland, July 2004.

