Building Ad-Hoc Clouds with CloudAgora

Tasos Bakogiannis, Ioannis Mytilinis, Katerina Doka and Georgios Goumas
Computing Systems Laboratory,
National Technical University of Athens, Greece
{abk, gmytil, katerina, goumas}@cslab.ece.ntua.gr

***Abstract*—The public Cloud market has become a monopoly, where a handful of providers - which are by default considered as trusted entities - define the prices, accumulate knowledge from users' data and computations and strengthen their already privileged position. As a remedy we propose CloudAgora, a platform that democratizes the Cloud market by allowing individuals and companies alike to compete on equal terms as potential resource providers, while enabling users to access low-cost storage and computation without having to blindly trust any central authority.**

During the demo, the attendees will be able to interact with CloudAgora through an easy-to-use UI, which will allow them to act both as users and as providers. As users, the attendees will have the chance to request storage or compute resources, upload data and outsource task processing over remote infrastructures. As providers, they will be able to participate in auctions, serve requests and offer validity proofs upon request. Moreover, the audience will experience first hand how the underlying blockchain technology is used to record commitment policies, publicly verify off-chain services and trigger automatic micropayments.

***Index Terms*—Blockchain, Cloud Computing, Marketplace, Verifiable Computation**

I. Introduction

In the last decade, Cloud Computing has become an essential part of most businesses, offering them a seemingly infinite pool of resources, from raw compute power to application functionality, on-demand and on a pay-as-you-go manner, alleviating the burden of acquiring and maintaining new hardware and software infrastructures.

Although Cloud Computing relies on fundamental principles of distributed systems, it remains centralized in its philosophy: Cloud services are mainly based on a handful of large providers that act as trusted entities for the transfer, storage and processing of user or company data. Thus, on the one hand the Cloud Computing market has become a monopoly, where a few big players determine price levels, which are non-negotiable and can be prohibitively high for deployments demanding specialized hardware. On the other hand, large Cloud providers accumulate vast amounts of data, getting a great head start in races like the ones of machine learning and big data processing.

Blockchain technology seems to be the key to solving the aforementioned issues, offering real decentralization, transparency and strong security guarantees based on proof rather than trust. Blockchain's distributed ledger can store data guaranteeing its validity and immutability, whereas smart contracts can execute pieces of code in a credible manner. However, Blockchains cannot be adopted as storage providers or computing engines per se, due to their limited computing and storage capacity, but rather as an enabling technology which keeps track of and validates off-chain operations. The challenge in this scenario is to find a secure way to guarantee the correctness of the off-chain service through the use of publicly verifiable proofs.

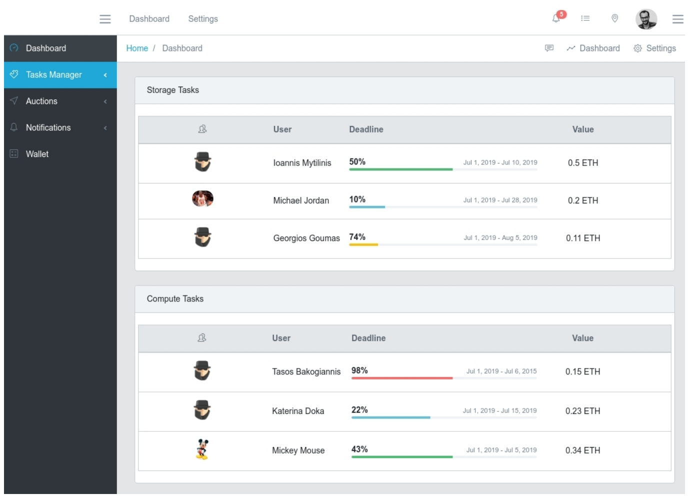
To that end we have proposed CloudAgora [1], a truly decentralized cloud that allows for on-demand and low-cost access to storage and computing infrastructures. The goal of CloudAgora is to create a blockchain-based platform where participants can act either as providers, offering idle CPUs and available storage, or as consumers, renting the offered resources and creating ad-hoc virtual cloud infrastructures. Storage and processing capacities are monetized and their prices are governed by the laws of supply and demand.

CloudAgora has been implemented as a dApp on top of Ethereum. This demo showcases the use of CloudAgora through an easy-to-use UI, which will allow attendees to act both as users, being able to request storage or compute resources, upload data and outsource computations over remote infrastructures, and as providers, serving requests.

II. CloudAgora Overview

CloudAgora is a platform that enables the ad-hoc creation of truly decentralized cloud infrastructures. CloudAgora users can act as both clients, requesting remote storage or computation, and providers, offering such resources. In a nutshell, clients express a request for storage or computation and potential providers, be it individuals, companies offering idle or under-utilized resources or large datacenters, place bids in an auction-style manner. The height of the potential providers' bids in combination with their reputation defines the auction winner(s). The agreement between providers and consumers is encoded as a smart contract, which allows for traceability of actions and automatic triggering of payments. While storage and processing per se is performed off-chain, the integrity and availability of stored data as well as the correctness of the outsourced computation are safeguarded through proper verification processes that take place on-chain.

The system is hierarchically structured in two layers:
The market layer, which comprises a set of algorithms that define participants' incentives and mechanisms for the regulation of prices. The creation of a new cloud job, the decision on price levels and the assignment to a specific provider all belong to the market layer. The CloudAgora market rules are enforced through a set of smart contracts that work on-chain.



Storage Tasks

	User	Deadline		Value
	Ioannis Mytilinis	50%	Jul 1, 2019 - Jul 10, 2019	0.5 ETH
	Michael Jordan	10%	Jul 1, 2019 - Jul 28, 2019	0.2 ETH
	Georgios Goumas	74%	Jul 1, 2019 - Aug 5, 2019	0.11 ETH

Compute Tasks

	User	Deadline		Value
	Tasos Bakogiannis	98%	Jul 1, 2019 - Jul 6, 2015	0.15 ETH
	Katerina Doka	22%	Jul 1, 2019 - Jul 15, 2019	0.23 ETH
	Mickey Mouse	43%	Jul 1, 2019 - Jul 5, 2019	0.34 ETH

Fig. 1. Screenshot of CloudAgora's UI. The Figure illustrates the Tasks Manager lists for a service provider.

The storage/compute layer, where actual cloud services are provided. This layer contains algorithms that can work both on- and off-chain and ensure the provably proper operation of the whole system. The contracts of this layer audit clients and providers and guarantee that none is making profit against the rules of the market.

A. *Market Layer*

This layer initially allows clients to express new tasks, storage or computational, defining information related to the tasks' duration and difficulty: For storage tasks the difficulty is implied by the dataset size and for computational tasks by the amount of required gas, when the code to be executed is converted to Ethereum Virtual Machine code. Upon task creation, a new auction for this task is launched and all interested parties are informed through an event emitted on the blockchain. Any potential provider can place bids until the auction expiration. The auction winner is finally defined by the height of her bid in combination with her reputation score. A reputation score is maintained and taken into account in the matching of tasks to providers, to protect clients from malicious providers who offer services in extremely low prices.

B. *Storage Layer*

This layer implements the decentralized storage service. At this point the client's storage task has been assigned to one or more storage providers, as a result of the auction, and a storage contract that binds them is created. The contract contains information about the involved parties, the duration of the task and the payment, including a collateral by both parties to discourage malicious behaviors. Moreover, the contract holds the Merkle tree root hash of the client's data, which is used as a means to verify the integrity of the stored data. Optionally, the client may decide to encrypt and/or erasure encode her data to guarantee confidentiality and recovery of data even in the event of failures.

The data is transferred in an off-chain manner from the client to the provider(s), where it is stored. While the contract is active, the client can request her data or check that her data is available and intact at any point in time. The former is performed off-chain, while the latter is performed either off-chain or on-chain, with the client challenging the provider by requesting a number of Merkle proofs [2] for specific data blocks. In case the provider fails to provide the corresponding proofs, she loses her collateral and her reputation.

After the contract expiration, the provider can automatically claim her payment, under the condition that she still possesses the data. This can be proven again by sending a number of Merkle proofs to the storage contract.

C. *Compute Layer*

This layer implements the compute service in CloudAgora. It is implemented as a truebit-like [3] game, where outsourced algorithms are executed off-chain and the blockchain is only used for correctness proofs. The outline of such a game is as follows: A *solver* is selected for executing the task based on the auction described in the Market Layer. The solver privately performs computations off-chain and only after completion reveals the solution on the blockchain. Simultaneously, any other member of the system can also compute the same task in private and act as a *verifier*. If a verifier agrees with the solution provided by the solver, the solver gets paid and the game stops. In case of a disagreement, the verifier can challenge the solver and an interactive proof takes place on-chain. If the solver proves to be malicious, the verifier receives the solver's reward and the solver loses the deposited collateral. If the verifier has triggered a false alarm, she is obliged to pay for the resources wasted due to game.

III. Demonstration Description

During the demonstration, attendees will have the chance to interact with CloudAgora both as cloud *clients* and *providers*. From a client's perspective, a comprehensive and friendly UI will allow them to upload real data and assign computational tasks to peer-members of a decentralized infrastructure. Data integrity and result validity control are supported in an any-time fashion. As cloud providers, attendees will be able to participate in auctions, serve requests and offer validity proofs upon request. In both cases, an Ethereum wallet is going to provide real-time billing information for the corresponding CloudAgora account. In the following, we delve into the details of each role and showcase all functionalities that our system's UI offers.

A. *Cloud Provider*

A user that has been registered as a provider can take on both computational and/or storage tasks. As soon as she logs in the system, an admin *Dashboard* becomes available. Through this Dashboard, the user has access to: (i) the *Tasks Manager*, (ii) the *Auctions*, (iii) the *Notifications* and (iv) the *Wallet*. These components are visible to the left side-bar of our UI as illustrated in Figure 1.

The Tasks Manager. Figure 1 shows a view of the Tasks Manager screen. The tasks a provider is assigned are organized in two distinct lists: one for the storage tasks and another one for the computational ones. For each task, we can see the identity of the client, the deadline of the task as well as the amount of money the provider will earn upon delivery.

The Auctions. This component provides an explicit view of all auctions a provider currently participates.

The Notifications. In CloudAgora, the actions a provider needs to take are mainly triggered asynchronously through an event-based notification system. Actions can be distinguished into three categories: (i) *auction actions*, (ii) *service actions* and (iii) *proof actions*. Whenever a provider needs to take an action of any kind, an event is emitted. The NodeJS client of our application captures this event and adds a new notification in the UI (top right corner of Figure 1). Received notifications can also be inspected in the corresponding component of the left side-bar.

We further analyze the distinct notification types. Whenever a new bid is placed in an auction where a provider participates, a notification arrives. By checking the newly arrived notifications, the provider can inspect the auction's details and decides whether to place a new bid or not.

Notifications about service actions are received upon winning an auction. These notifications inform the provider that she needs to receive the data/code as required per task.

Finally, a provider may also receive a message in case a cloud client desires to control the integrity of its data. Upon receiving such a request, the provider goes to the page with the details of the specific task and clicks on the "Validate" button. An on-chain Merkle proof is then instantly created and a blockchain event is emitted to inform the user on the results of the validation process.

The Wallet. By clicking on this component of the left side-bar, details about the Ethereum wallet of the provider appear on the screen. In this page, the user can inspect the balance of her account as well as her recent blockchain transactions.

B. Cloud Client

When a user logs in as a cloud client, she faces a UI very similar to the one we discussed for service providers. A client can create new computational and/or storage tasks and assign them to one or more providers. The task assignment is carried out through the auction mechanism of CloudAgora. Apart from creating new tasks, our UI also enables the management of existing ones: operations such as asking for validity proofs or canceling tasks are just a few clicks away.

In a typical scenario, a client launches the application and navigates to the *Tasks Manager* similarly to the provider's case. For creating a new task, the client has to specify its type (i.e., storage or computation), and the deadline for the auction to take place. Depending on the task's type, some additional information may be required. For *storage tasks* the client also has to specify the size of the file to be stored remotely, the duration of the *storage contract*, as well as the number of providers that are going to store parts of the file. In the case of a *computational task*, the extra information needed to be specified is the amount of gas required to execute the computation. Once the task is created an auction managed by an Ethereum contract is initiated. The progress of the auction is tracked in real-time and the client can monitor it in the corresponding tab of our user interface.

When the auction is successfully finalized, the *auction contract* creates a storage/compute contract that binds the client with the winning providers. As soon as the storage/compute contract is created, a new item appears in the corresponding list in the Tasks Manager interface. By clicking on that item, the client can view and manage the details of the corresponding contract. In general, the management of a task involves (i) uploading data/code, (ii) tracking its progress, (iii) requesting validity proofs and (iv) ordering for its termination. In the next two paragraphs, we describe in more detail these operations for the cases of both storage and computational tasks.

In the case of storage, from the contract's page the client has the ability to serve the file to the providers. Specifically, by selecting the file to be served, the application performs erasure encoding and splits the file in equal parts, one for each provider. The client can inspect the status of a contract at any time. Moreover, she can challenge a provider on the validity of a specific block. The provider under challenge has to prove she still has the file at her possession and that data has not been tampered. Finally, at any time, the client can download the file and conclude the task by finalizing the contract.

In the case of a computational task, the same interface provides to the client the means to serve the executable to the provider and monitor the stages of the computation. Nevertheless, in the case of such a task, the client cannot directly challenge the provider but the verification process is based on a truebit-like game.

Acknowledgments

This research is co-financed by Greece and the European Union (European Social Fund- ESF) through the Operational Programme "Human Resources Development, Education and Lifelong Learning 2014-2020" in the context of the project "Data Sovereignty through the use of Blockchain"(MIS 5004883).

References

[1] K. Doka, T. Bakogiannis, I. Mytilinis, and G. Goumas, "Cloudagora: Democratizing the cloud," in *International Conference on Blockchain*. Springer, 2019, pp. 142–156.

[2] R. C. Merkle, "A digital signature based on a conventional encryption function," in *Advances in Cryptology - CRYPTO '87*. Springer Berlin Heidelberg, 1987, pp. 369–378.

[3] J. Teutsch and C. Reitwießner, "A scalable verification solution for blockchains," *URL: https://people. cs. uchicago. edu/teutsch/papers/truebit pdf*, 2017.