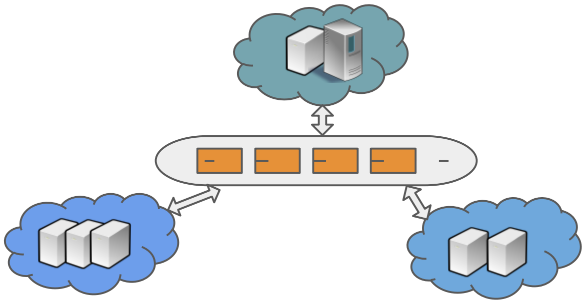
Towards a Democratic Federation for Infrastructure Service Provisioning



Sandip Chakraborty

Department of Computer Science & Engineering

Indian Institute of Technology Kharagpur

CNeRG

Complex Network Research Group – CNeRG@CSE, IITKGP

CNeRG Members

- 7 Faculty Members, ~35 Research Scholars

						
Niloy Ganguly Complex and Social Networks	**Animesh Mukherjee** Complex and Social Networks	**Bivas Mitra** Complex Networks, Systems	**Pawan Goyal** Natural Language Processing	**Sourangshu Bhattacharya** Machine Learning	**Sandip Chakraborty** Computer Systems and Networks	**Saptarshi Ghosh** Information Retrieval, Social Computing

Group Overview

- The group started in 2005

- Around 25 research scholars graduated
  - Placed in academics as well as in research labs
  - Postdocs in several institutes world-wide

- Major awards won by the PhD students
  - Yahoo Key Scientific Challenge Honorable Mention 2011
  - Microsoft Techvista Best Poster Award 2013, 2015
  - XRCI Best Student Thesis Award 2015
  - INAE Best Student Thesis Award 2016
  - IBM Best Student Thesis Award 2016

CNeRG@Web

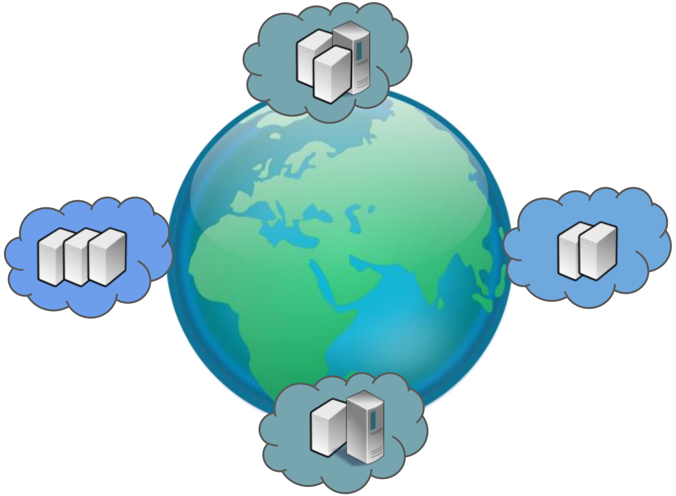
http://www.cnergres.iitkgp.ac.in/

@cnerg

Bishakh Chandra Ghosh, Sourav Kanti Addya, Anurag Satpathy, Soumya K. Ghosh and Sandip Chakraborty, “**Towards a Democratic Federation for Infrastructure Service Provisioning**”, in proc. of the 2019 IEEE International Conference on Services Computing (IEEE SCC 2019), Milan, Italy, July 08-13 2019

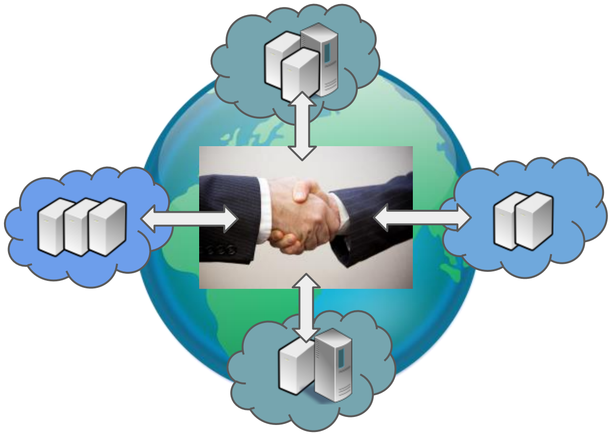
Cloud Federations

Collaboration among different **Cloud Service Providers** (CSPs), whereby they agree to mutually share their own resources for their overall benefit.



Cloud Federations

Collaboration among different **Cloud Service Providers** (CSPs), whereby they agree to mutually share their own resources for their overall benefit.

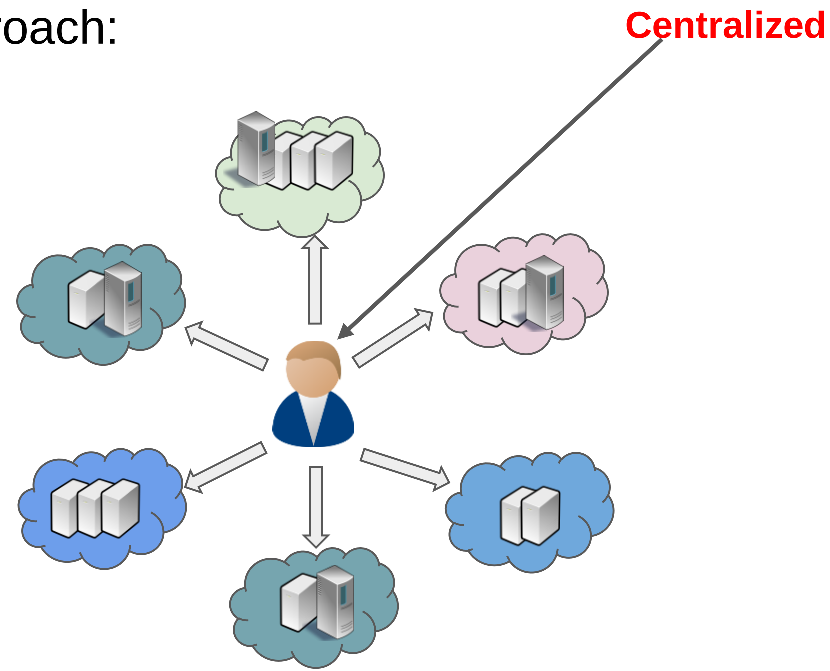


Motivation for cloud federation

- **Sharing** of computing resources.
- **Aggregation of unused resources** from different service providers.
- Bringing services closer to customers by maximizing the **geographical dispersion**.
- **Tackling data protection laws** that requires data to be stored within country's boundary.

Existing cloud federations

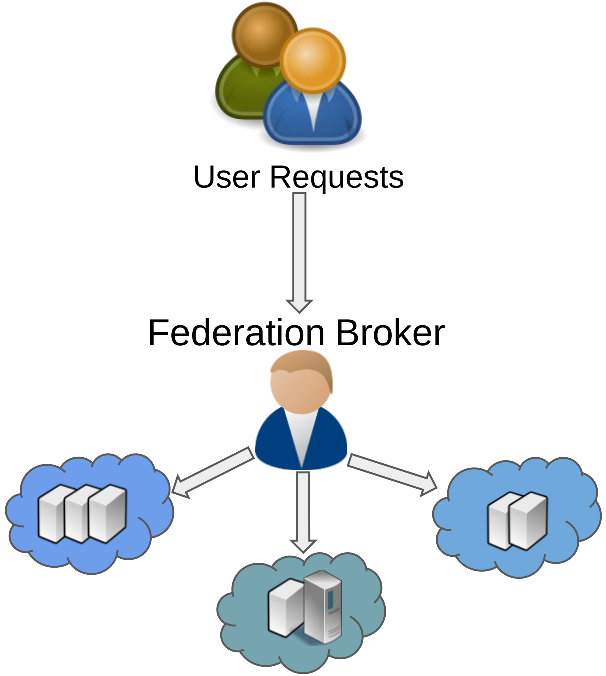
Mostly **Centralized** Approach:



Existing cloud federations

Mostly **Centralized** Approach:

1. Centralized broker



Existing cloud federations

Mostly **Centralized** Approach:

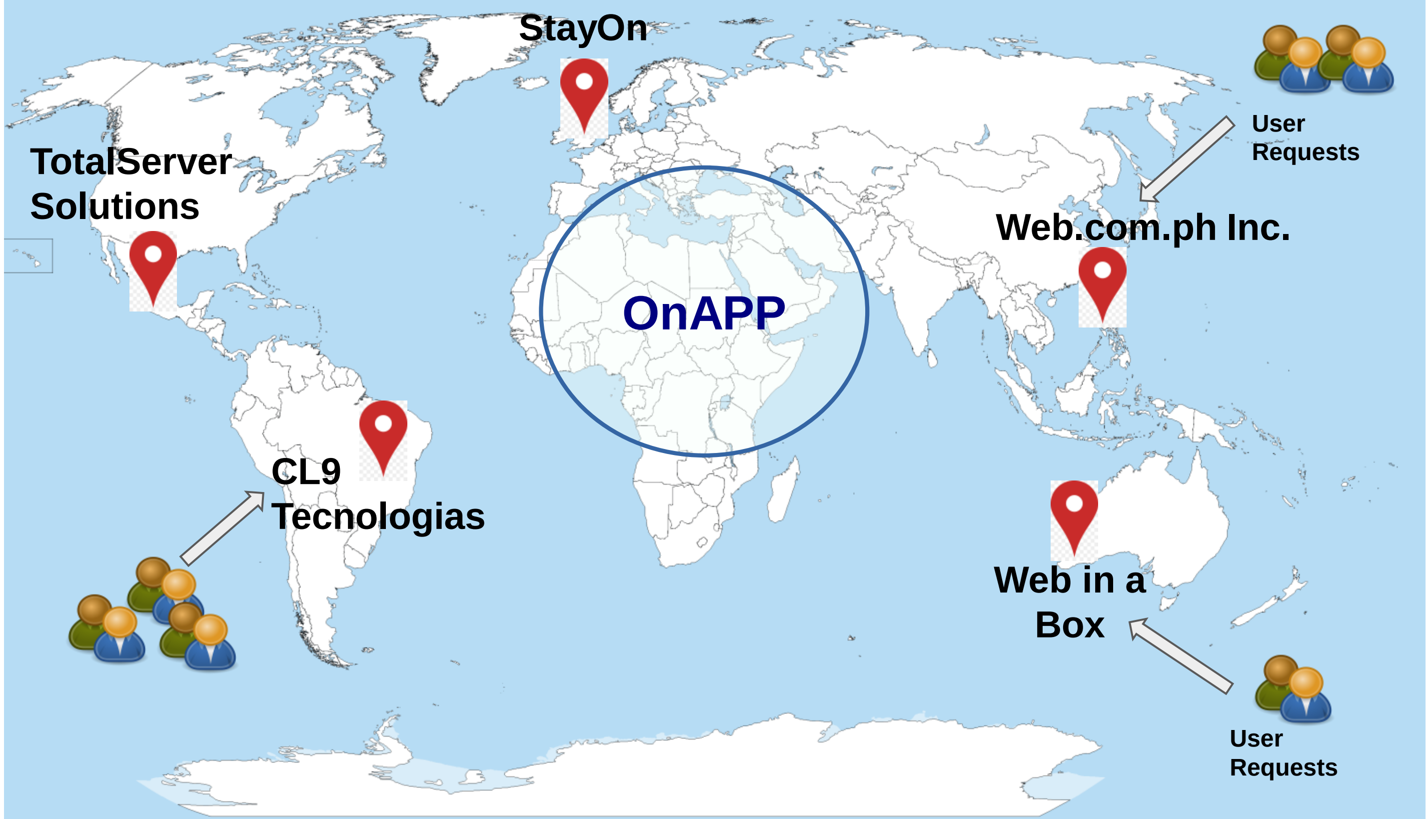
2. Centralized exchange

Existing cloud federations

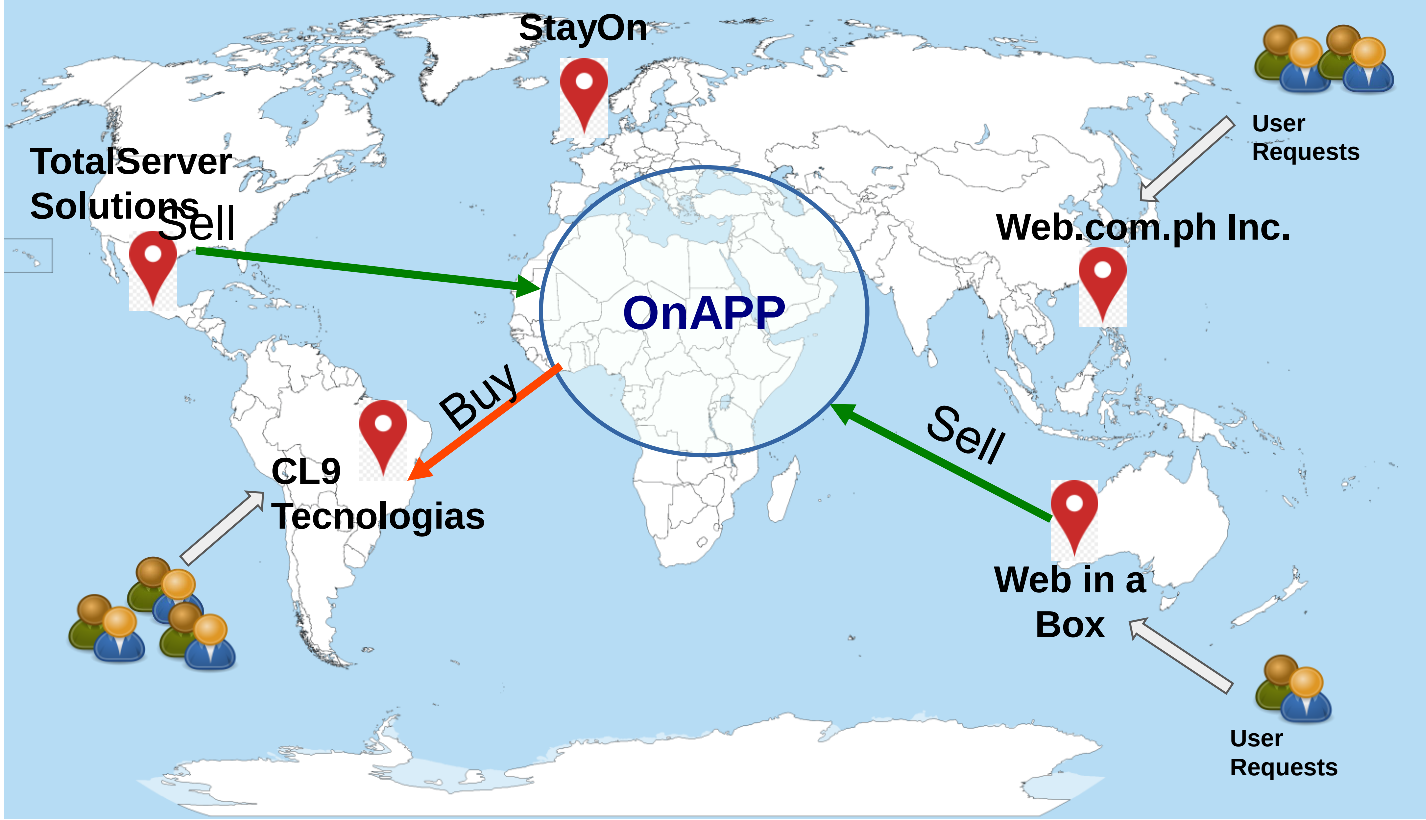
Existing cloud federations

Existing cloud federations

Existing cloud federations

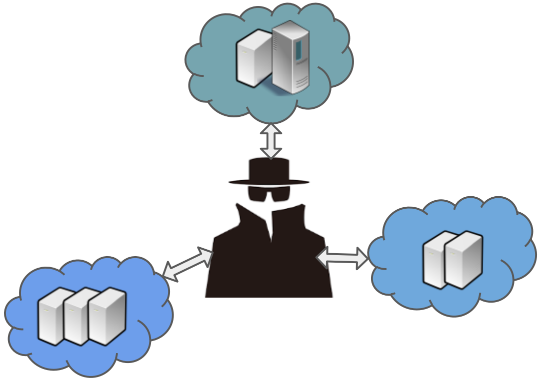


Existing cloud federations



Limitations of existing federations:

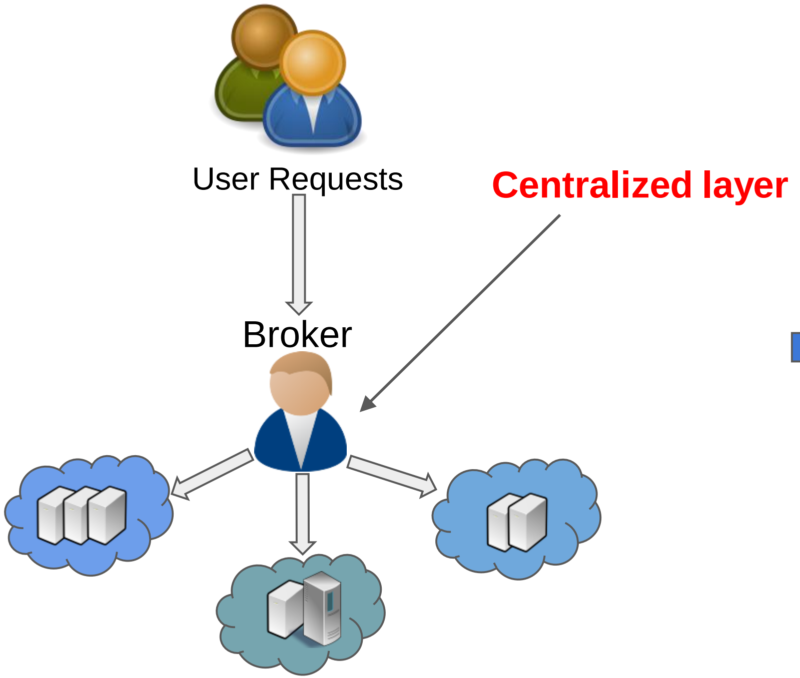
1. Profit sharing with central broker
2. Biasness of broker towards certain service providers
3. Price manipulation (*Broker can be malicious*)
4. Unfair dispute resolution
5. Central point of failure



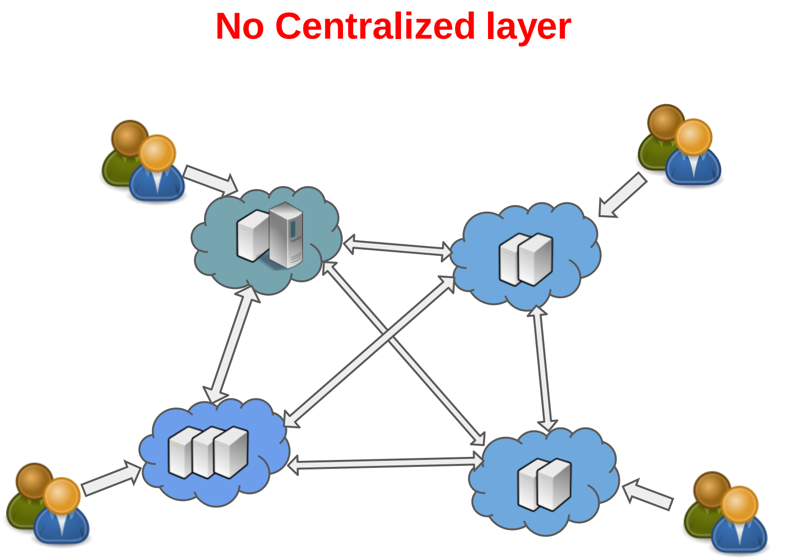
Objective

Remove the central broker **and design a transparent distributed system for cloud federation.**

Centralized to Decentralized



Centralized Cloud federation



Decentralized Cloud federation

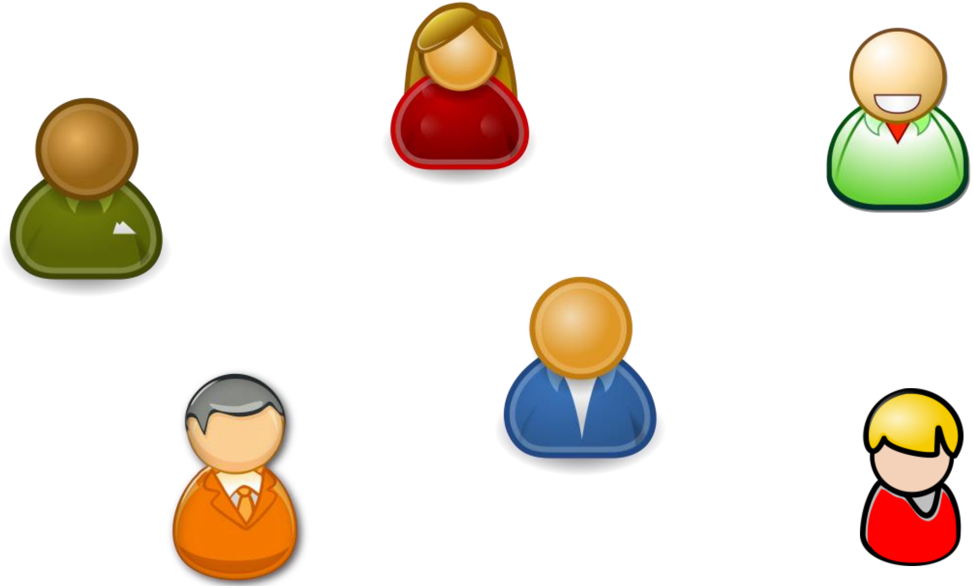
Challenges

- A decentralized platform for exchange of infrastructure resources (VM) must be developed.
- The system must allow coordination between service providers while enforcing FLA, without the help of any broker.
- Cloud functions such as VM Placement and VM Migration needs to be coordinated over the decentralized architecture.
- Fair ordering of transactions must be ensured

Blockchain

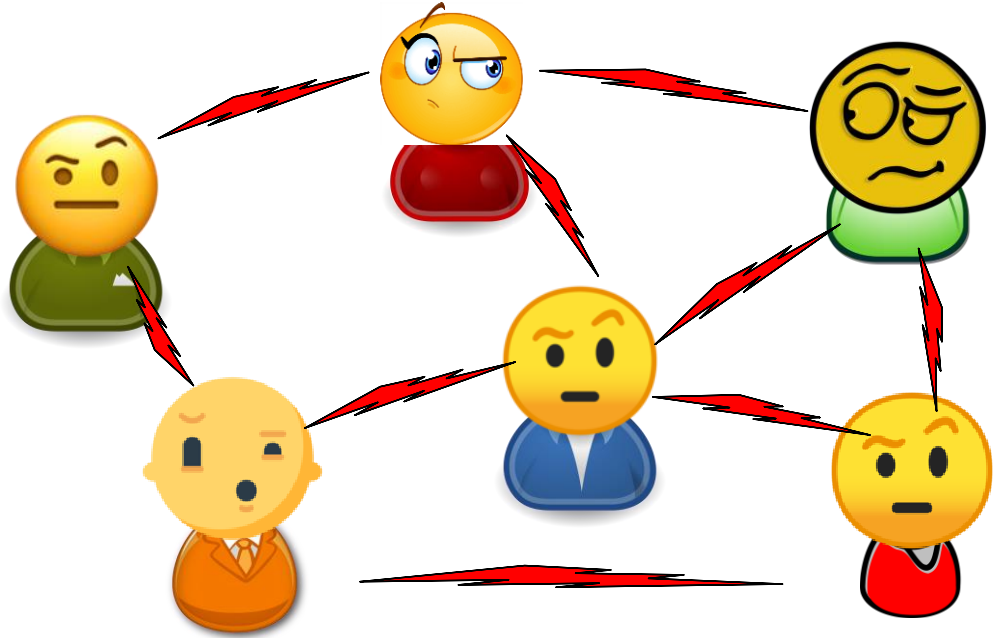
Blockchain

- **Multiple authoritative domains**



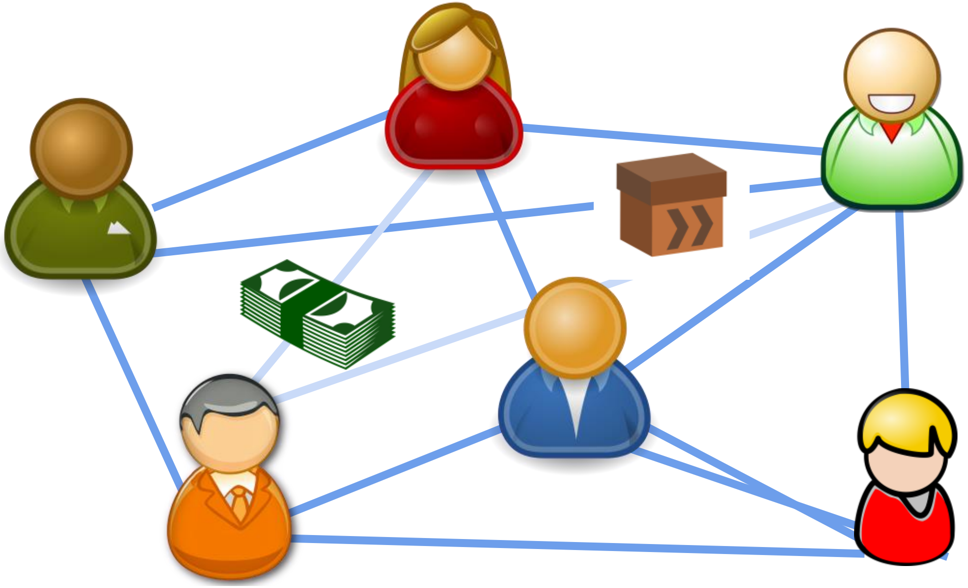
Blockchain

- Multiple authoritative domains
- **Do not trust each other**



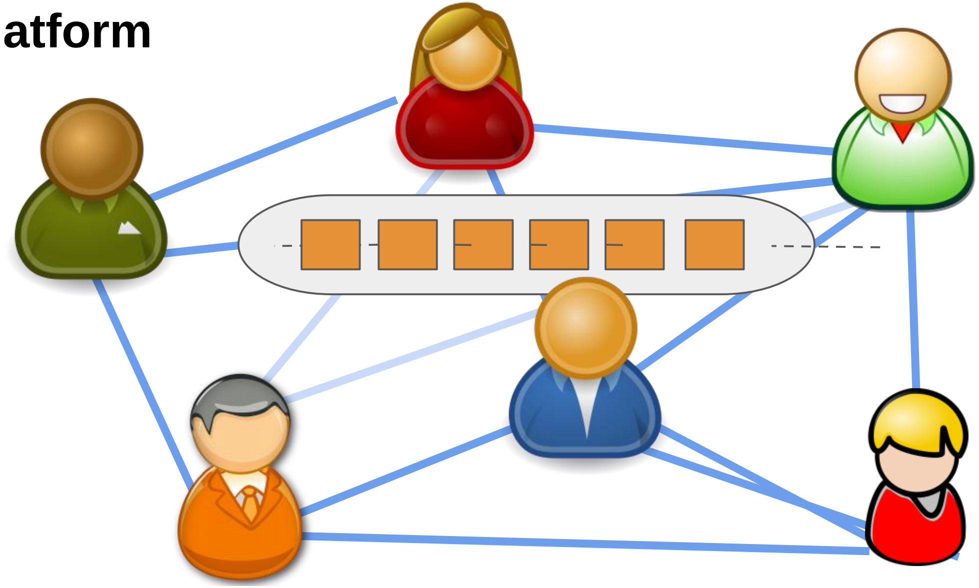
Blockchain

- Multiple authoritative domains
- Do not trust each other
- **Can collaborate for mutual benefit**



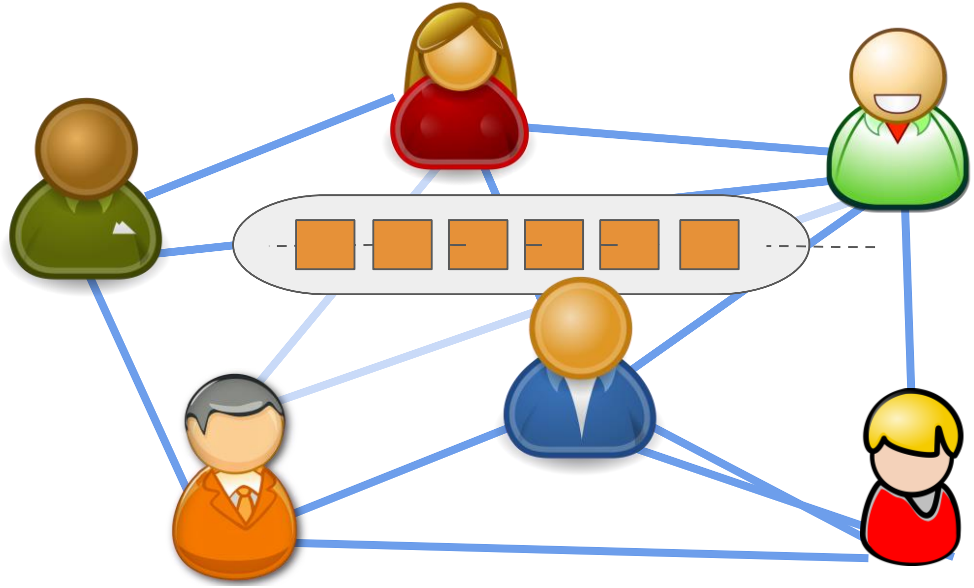
Blockchain

- Multiple authoritative domains
- Do not trust each other
- Can collaborate for mutual benefit
- **Blockchain provides such trustless decentralized platform**



Blockchain

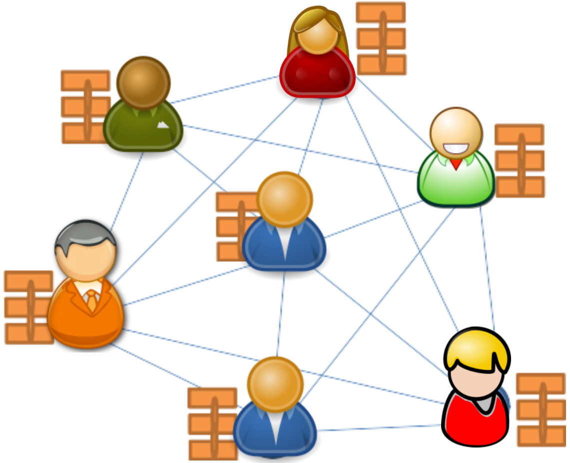
A **decentralized** computation and information sharing platform that enables **multiple authoritative domains**, who **do not trust each other**, to cooperate, coordinate and collaborate in a rational decision making process.



Blockchain

Essentially a decentralized database with strong consistency support.

- Every node maintains a local copy of the global data.
- The system ensures consistency among the local copies.
- The local copies at every node is identical.
- The local copies are always updated based on the global information (consensus).



Blockchain

Blockchains work like a public ledger - a database of historical information

Some important aspects:

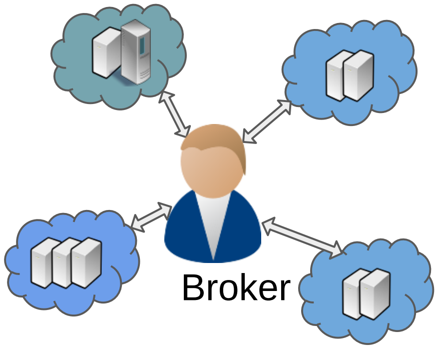
Protocols for Commitment: Ensure that every valid transaction from the clients are committed and included in the blockchain within a finite time.

Consensus: Ensure that the local copies are consistent and updated.

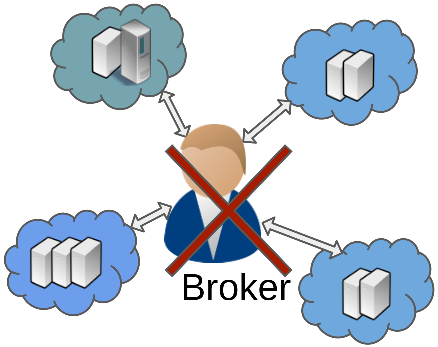
Security: The data needs to be tamper proof. Note that the clients may act maliciously or can be compromised.

Privacy and Authenticity: The data (or transactions) belong to various clients; privacy and authenticity needs to be ensured.

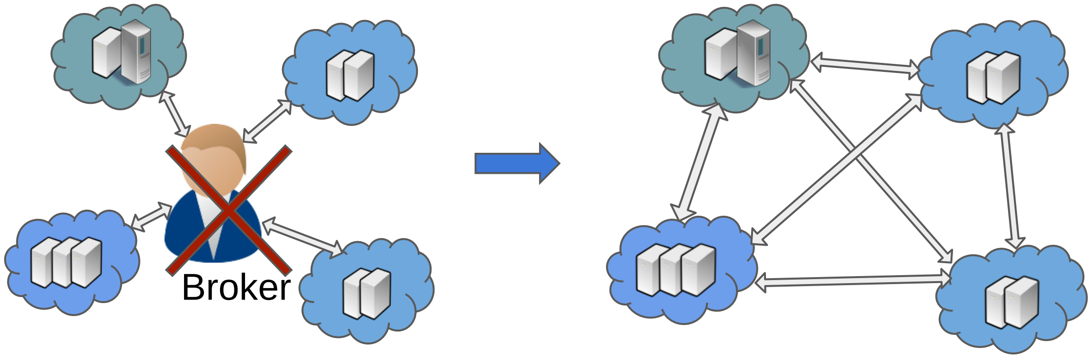
Proposed Architecture



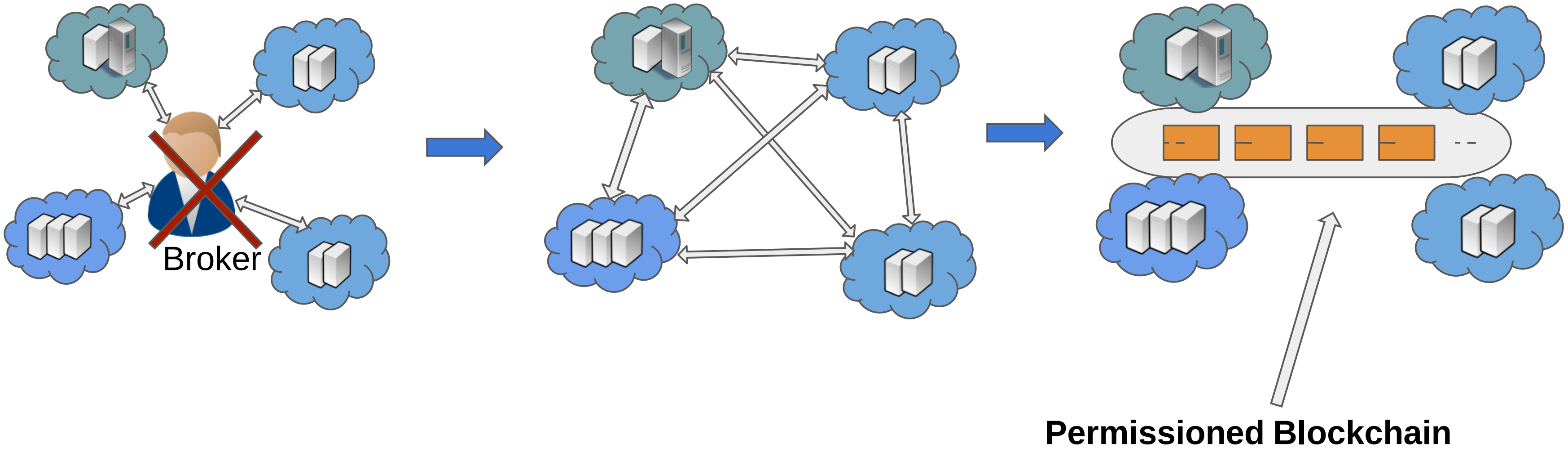
Proposed Architecture



Proposed Architecture



Proposed Architecture



Proposed Architecture

Permissioned blockchain based **decentralized exchange** for **democratic** cloud federations: ***CloudChain***

Proposed Architecture

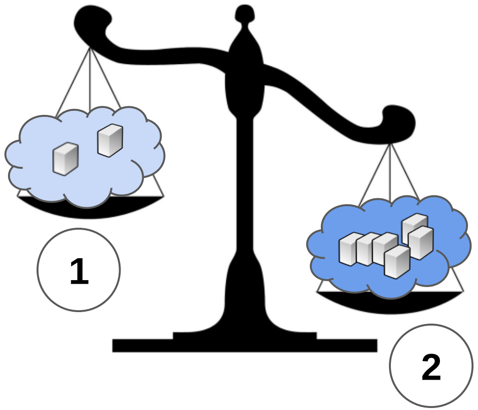
We consider that the federation contains two types of service providers namely,

1. Demanding service providers:

   Suffer from **resource limitations** and require other members of the federation to create instances for them at peak loads.

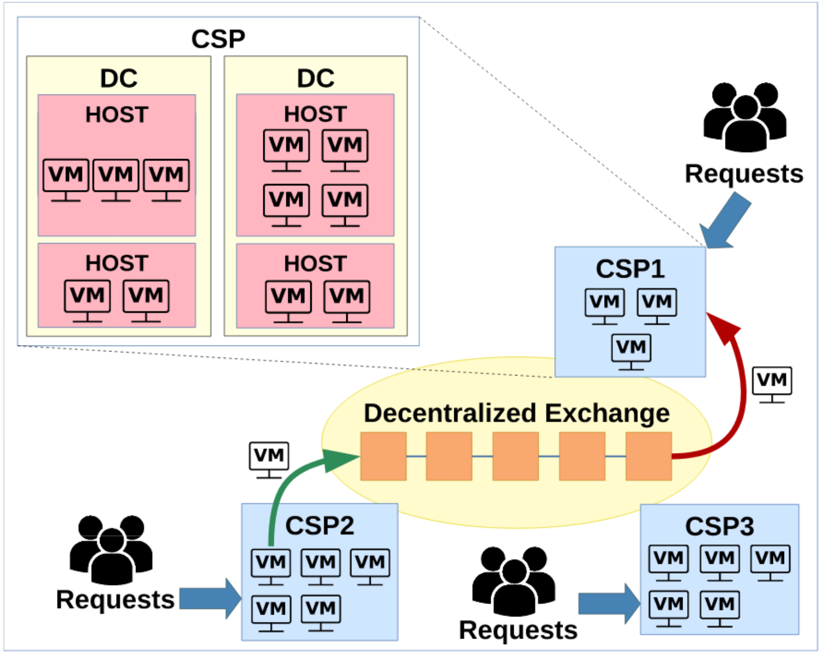
1. Supplying service providers:

   Having **abundant resources** which goes unused.

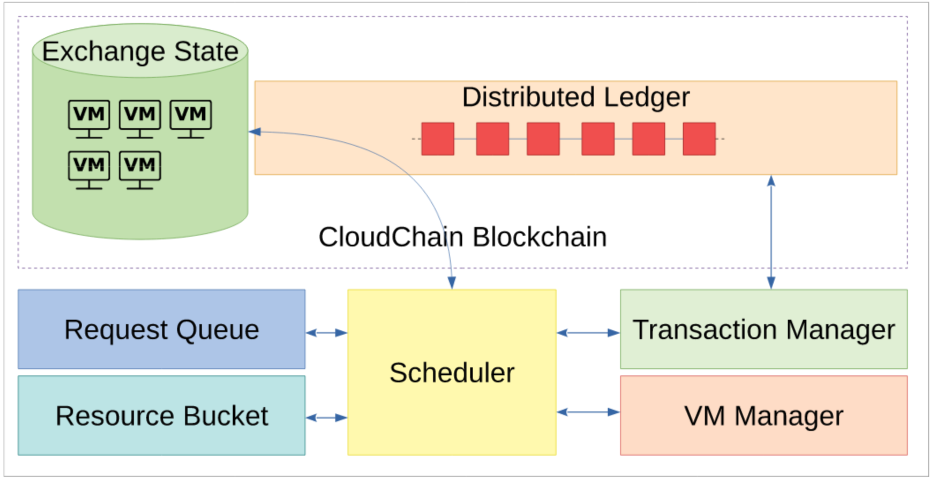


Proposed Architecture

CloudChain



CloudChain model overview



Components of *CloudChain*

Components

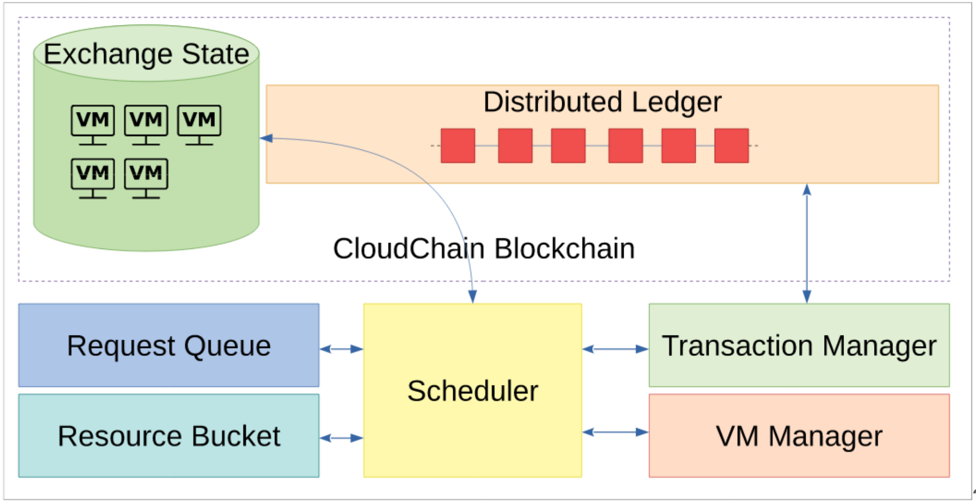
1) *CloudChain* Blockchain: Distributed Ledger & Exchange State

2) Request Queue and Resource Bucket

3) Scheduler

4) Transaction Manager

5) VM Manager



Components

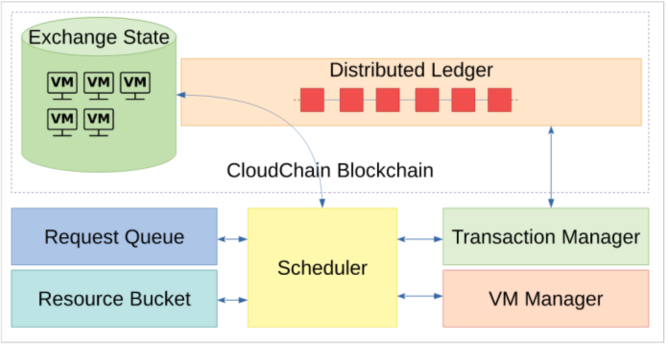
Request Queue (*ReQ*): queue of incoming multi-tier web application requests

Resource Bucket:

Bucket of available resources, which may include both local resources and exchange resources.

1) Local resource bucket (*ResBlocal*)

2) Exchange resource bucket (*ResBexchange*)

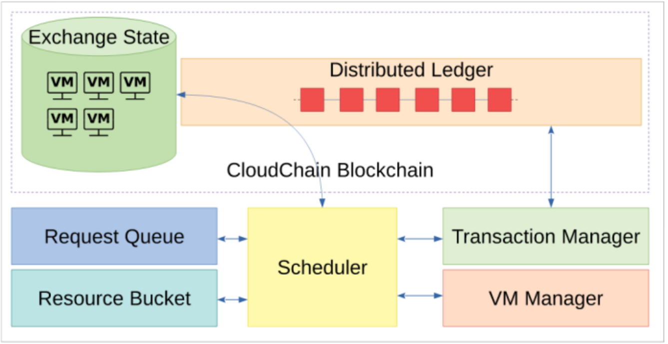


Components

Scheduler: Coordinates all the components of *CloudChain*

Transaction Manager: An interface to the *CloudChain* Blockchain.

VM Manager: Manages creation, deletion and access to VMs.



***CloudChain* Blockchain** serves as an information registry that maintains the current state of available resources and demand patterns. Thus it acts as a common marketplace where different CSPs can offer their unused or excess resources for outsourcing, and rent resources from other CSPs when required.

Exchange State

Offerings			
ID	Supplying SP	Specs	Price/BTU
o_1	Cloud 1	2 Cores, 4GB, UK	15$/Month
o_2	Cloud 2	2 Cores, 8GB, US	18$/Month
o_3	Cloud 1	2 Cores, 2GB, UK	0.16$/Day

Requests			
ID	Demanding SP	Offering Id	Duration
r_1	Cloud 2	o_1	6 Months
r_2	Cloud 3	o_2	12 Months

Associations			
ID	Demanding SP	Offering Id	Duration
a_1	Cloud 3	o_5	3 Months
a_2	Cloud 5	o_4	7 Days

CloudChain Blockchain

The high level **operations** that the CSPs can perform on the exchange are:

1) **Offer** a new resource

2) **Modify** an existing offering

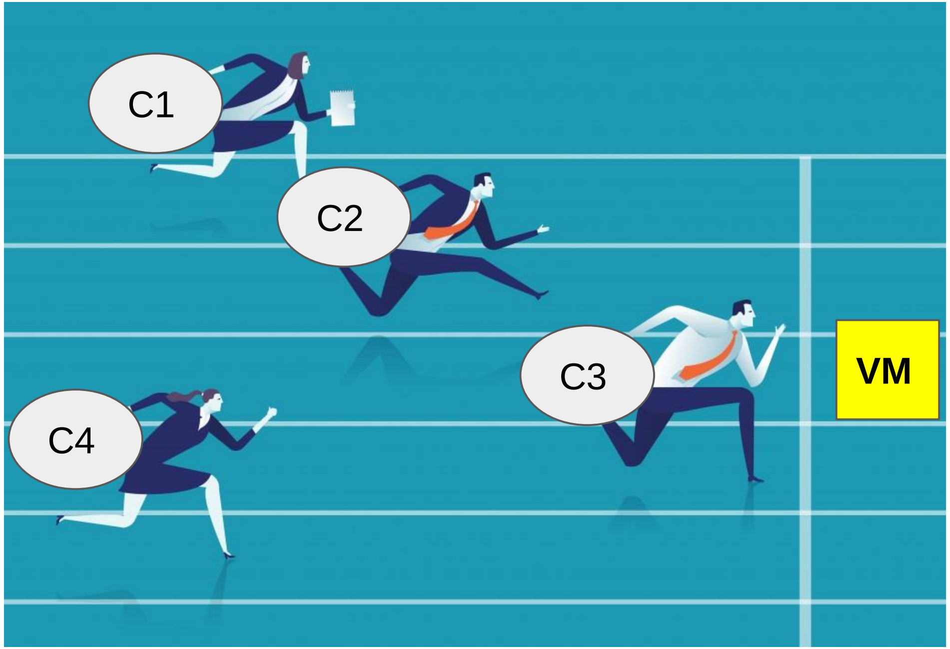
3) **Query** for available resources offerings

4) **Request** to rent a resource

5) **Grant/Reject** a request

Fair Ordering

- Different CSPs may be competing for the same resource in the exchange.
- Whoever makes the request first essentially wins the resource.
- *Therefore, CloudChain* needs to ensure fairness in the ordering of events.



Fair Ordering

- Ordering in a centralized system is trivial.
- The events are ordered in the order of the messages received by the central orderer.
- However the central orderer may be malicious or biased and **tamper the ordering.**
- In a decentralized setting, ordering is difficult as there is no trusted global clock or order.

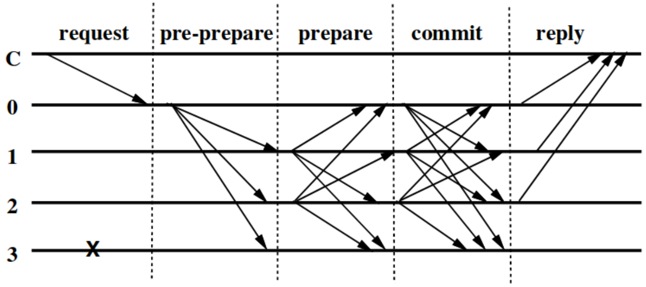
Fair Ordering: PBFT

- In well known BFT protocols like **PBFT**[5], ensures total ordering of requests, deterministic execution and liveliness.
- However, the primary decides the ordering of the requests.
- The primary is the single point of contact for the proposing clients, so it might be cleverly malicious and forward the requests of its preferred clients, by delaying others' requests.

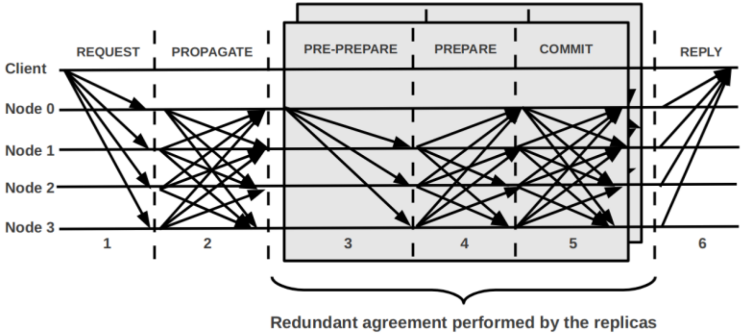
[5] Castro, Miguel, and Barbara Liskov. "Practical Byzantine fault tolerance." OSDI. Vol. 99. 1999.

Fair Ordering: RBFT

- **RBFT**[6] uses similar same three phase protocol is used as in PBFT.
- But instead of one primary replica executing one instance of the protocol, f+1 protocol instances are run in parallel, each with a different primary.



PBFT



RBFT

[6] Aublin, Pierre-Louis, Sonia Ben Mokhtar, and Vivien Quéma. "Rbft: Redundant byzantine fault tolerance." 33rd International Conference on Distributed Computing Systems. IEEE, 2013.

Fair Ordering: RBFT

- Out of f+1 instances, one is master instance and rest f are backup instances.
- Each node monitors the throughput of the f+1 instances.
- If 2f + 1 nodes observe that the throughput of the master instance is lower than a given threshold as compared to backup instances, then the primary of the master instance is considered to be malicious.
- A view change is triggered and a new primary is chosen.
- RBFT also monitors latency of requests from different clients.
- Thus, RBFT achieves robustness and also fairness to some extent.

Fair Ordering: Modified RBFT

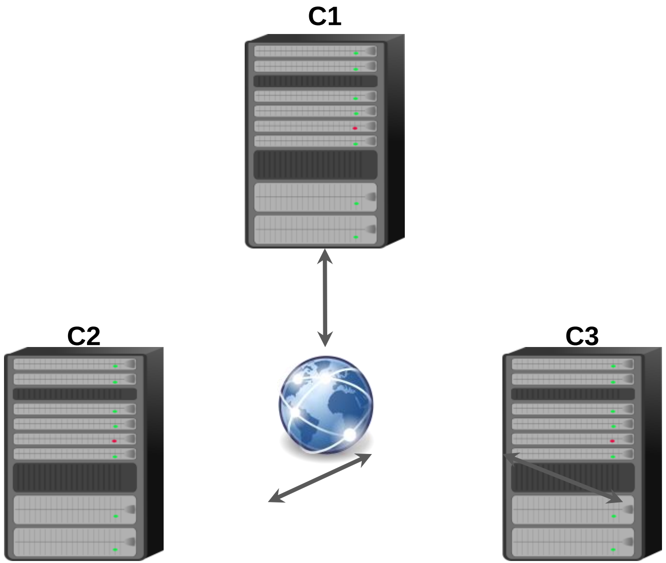
- Our proposed consensus protocol modifies RBFT to **monitor fairness in ordering of the requests**.
- Each node monitors the ordering of the f+1 instances and compares the ordering of the master instance with the backup ones.
- If 2f+1 nodes observe that the ordering of the master instance is ***N*** edit distance away from that of the backup instances, then a view change is triggered. (***N*** is a configurable parameter)

Testbed Setup

- 3 Hosts, each acting as a cloud connected over the network.

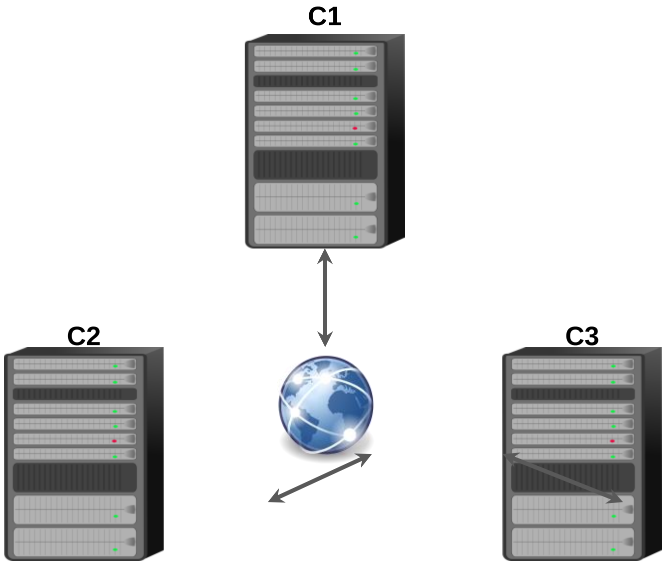
Testbed Parameters.

Category	Value
Number of CSP	3
Number of DC per CSP	3
$\mathcal{C}_1$ Config	2.7 GHz, Intel Xeon(R) 48 core, 256 GB Memory
$\mathcal{C}_2$ Config	3.2 GHz, Intel Core i5 4 core, 20 GB Memory
$\mathcal{C}_3$ Config	2.7 GHz, Intel Core i3 4 core, 8 GB Memory
Containerization	Docker 18.06
Language used	Go 1.10, Python 2.7



Testbed Setup

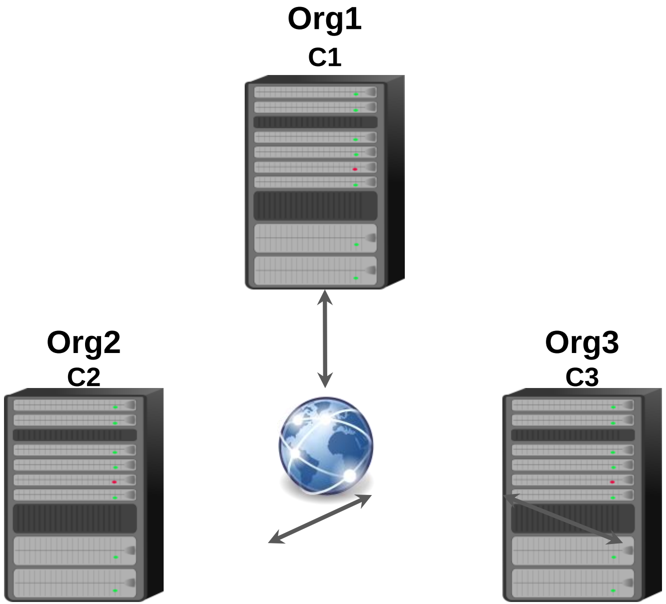
- 3 Hosts, each acting as a cloud connected over the network.
- Hyperledger fabric for blockchain (v1.3.0)



HYPERLEDGER FABRIC

Testbed Setup

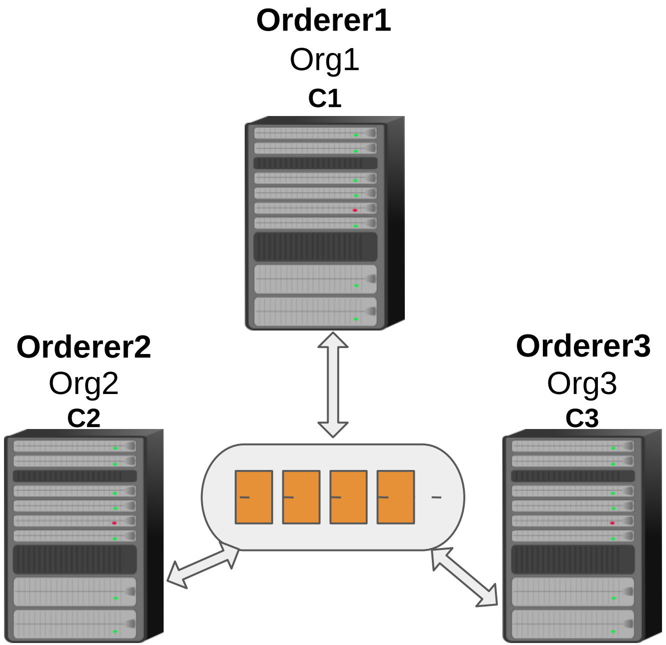
- 3 Hosts, each acting as a cloud connected over the network.
- Hyperledger fabric for blockchain (v1.3.0).
- Each cloud belongs to a separate **organization**, and runs a **peer.**



HYPERLEDGER FABRIC

Testbed Setup

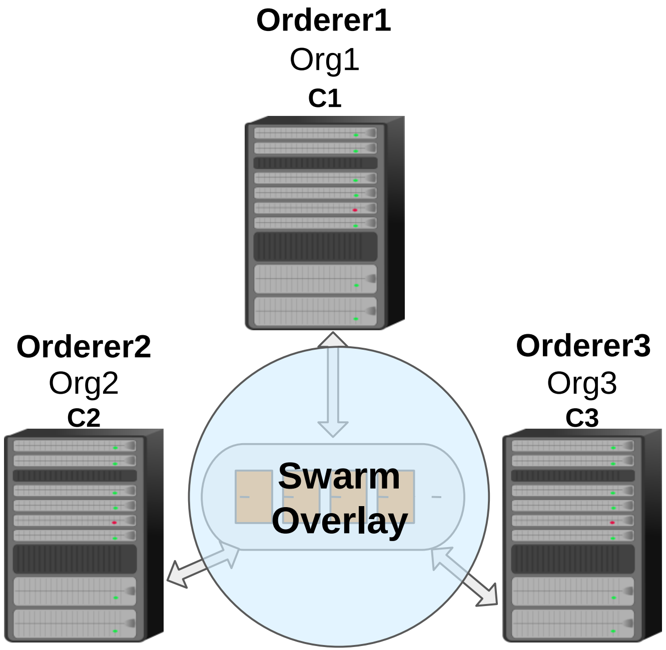
- Each cloud runs its own **orderer**.



HYPERLEDGER FABRIC

Testbed Setup

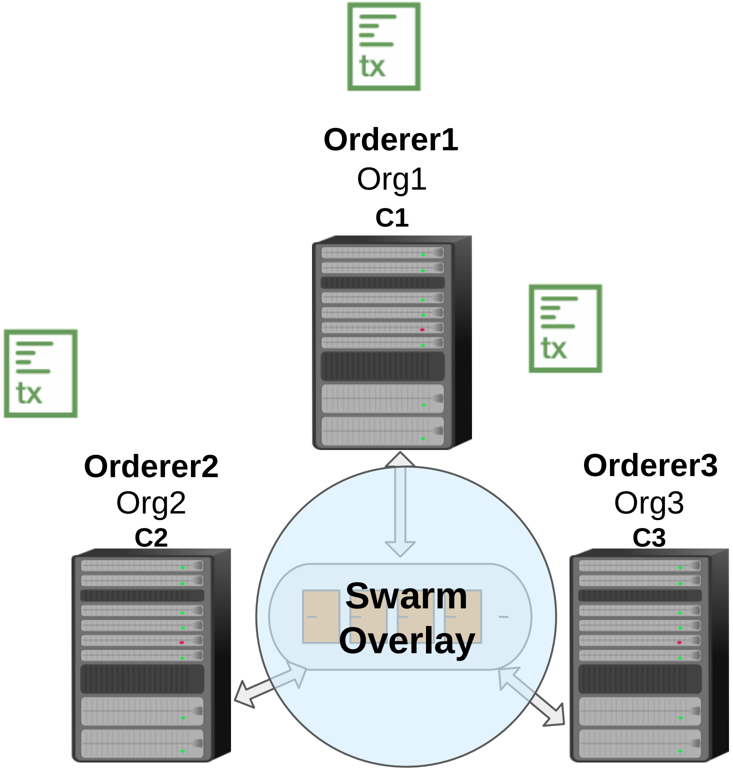
- Each cloud runs its own orderer
- Create a **docker swarm**.
- Create **overlay network**.



HYPERLEDGER FABRIC

Testbed Setup

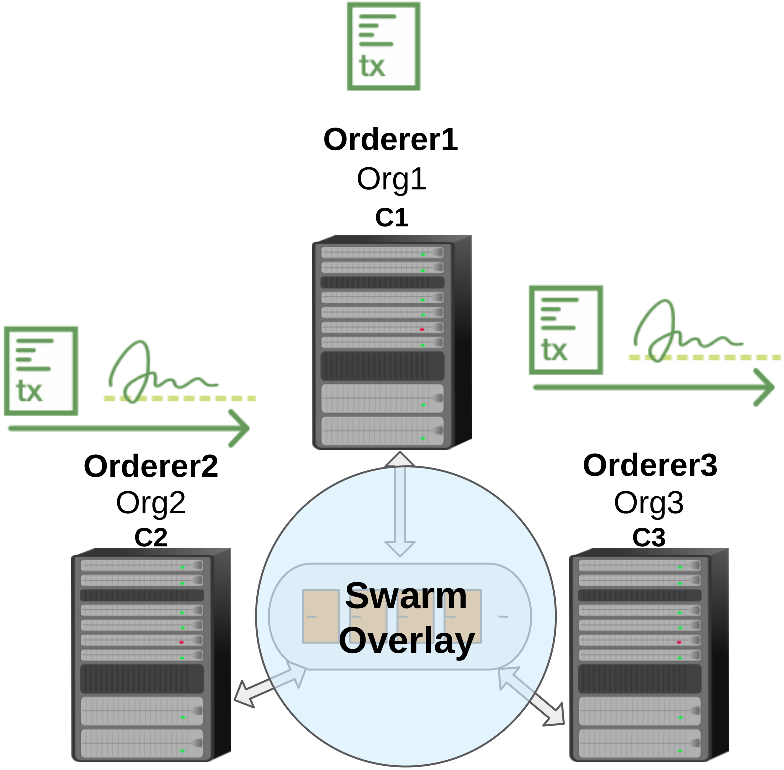
- Each cloud runs its own orderer.
- Create a **docker swarm**.
- Create **overlay network**.
- **Chaincodes** for *CloudChain* logic.



HYPERLEDGER FABRIC

Testbed Setup

- Each cloud runs its own orderer.
- The orderers use **BFT** protocol.
- **Chaincodes** for Cloud Exchange logic.
- **Endorsement policy** requiring the endorsement of the concerned demanding SP, supplying SP and the majority of other endorsing peers.

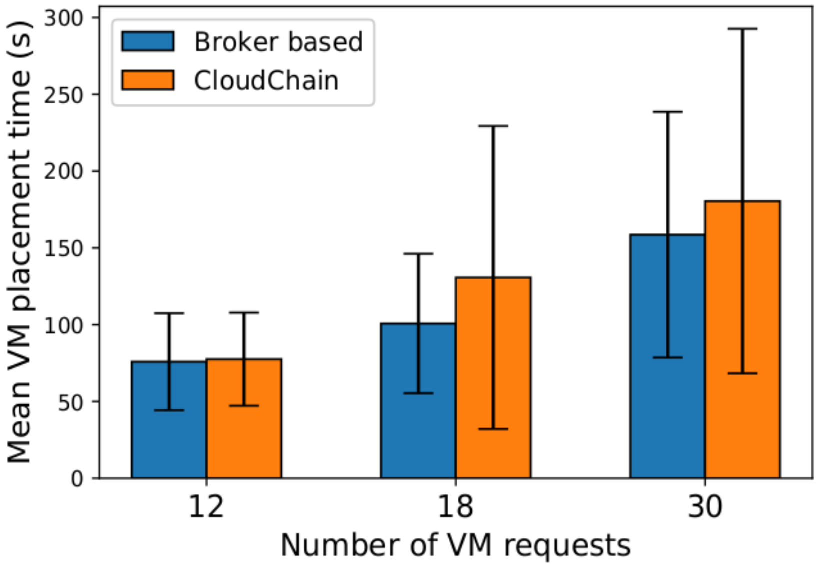


HYPERLEDGER FABRIC

Results

- Mean VM placement time in broker based federation and *CloudChain*
- Three scenarios
- Each CSP receives 4, 6, and 10 VM requests in first, second and third scenarios respectively.
- In case of broker based federation all the requests arrive at the broker first.

Mean VM placement time

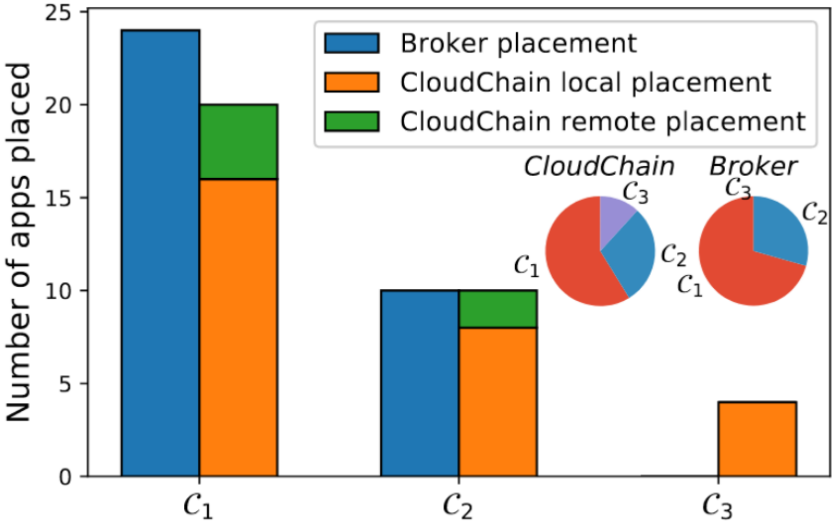


Very little compromise in performance

Results

- 34 multi-tier application requests
- *C1*, *C2* and *C3* receiving 16, 8 and 10 requests respectively.
- In case of broker based federation, all requests arrive at the broker

Distribution of user requests across different CSPs



C3 is starved in case of broker based system

***CloudChain* shows fair distribution**

Conclusion

***CloudChain* over Federation brokers:**

Conclusion

***CloudChain* over Federation brokers:**

- **Decentralized**

- *CloudChain* is not owned by any single governing body.
- Instead it is collaboratively maintained by all the participating CSPs.
- Removes any central point of failure
- Ensures distributed and democratic control.

Conclusion

***CloudChain* over Federation brokers:**

- Decentralized
- **Transparent**

- In broker based system, all the communications between a CSP and the broker remains confidential.
- In contrast, every transaction and every operation on the *CloudChain* platform is shared among the CSPs.
- The process of updation of any information is transparent and based on mutual agreement.

Conclusion

***CloudChain* over Federation brokers:**

- Decentralized
- Transparent
- **Autonomy**

- In broker based federations the requests come at the broker.
- The broker is in full control of the allocation of the requests among the participating CSPs.
- In *CloudChain*, each CSP functions with full autonomy as an individual service provider.
- It can insource/outsource resources according to its own policy

Conclusion

***CloudChain* over Federation brokers:**

- Decentralized
- Transparent
- Autonomy
- **Immutable**

- The distributed ledger of the *CloudChian* blockchain keeps history of every operation on the exchange.
- Each of these operations are accepted and agreed upon by the majority of the participating CSPs.
- In case of any dispute, this immutable log can be audited and a decision can be taken accordingly.

Conclusion and Future work

***CloudChain* over Federation brokers:**

- Decentralized
- Transparent
- Autonomy
- Immutable
- **Fairness**

- Autonomous process of resource exchange among the CSPs through *CloudChain* ensures fair distribution and profit sharing.
- No single authority can dictate the allocation of requests among the CSP.
- Fair ordering of events further ensures the fairness of the system.

A Few Open Research Problems

- Support for live VM migration
  - Who will initiate the migration?
  - How the migration from one CSP to another CSP will be controlled?
  - How the pricing and billing will be maintained?
- FLA monitoring
  - How can you detect adversaries in the FLA?

Complex Networks Research Group (CNeRG),

Department of Computer Science and Engineering,

IIT Kharagpur

INDIA 721302

http://www.cnergres.iitkgp.ac.in/