Cloud Based Storage System

CLOUD COMPUTING EXAM

Marco Tallone

September, 2024

SDIC Master Degree

UniTS

Introduction: Exam Exercises

Exercise 1 — CLOUD BASIC

Cloud Based Storage System with Docker Compose

Exercise 2 — CLOUD ADVANCED

Cloud Based Storage System with Kubernetes

Exercise 3 — CLOUD ADVANCED

OSU Latency Test on Kubernetes Cluster

Exercise 1 Assignment

- **User authentication 🔑 and file management 📁**
  - ☐ Log in and log out
  - ☐ Upload, download, read and delete files
  - ☐ Private and shared files

Exercise 1 Assignment

- **User authentication 🔑 and file management 📁**
  - ☐ Log in and log out
  - ☐ Upload, download, read and delete files
  - ☐ Private and shared files
- **User administration 👮 and authorization 🔒**
  - ☐ User roles: admin and user
  - ☐ Admin can manage users

Exercise 1 Assignment

- **User authentication 🔑 and file management 📁**
  - ☐ Log in and log out
  - ☐ Upload, download, read and delete files
  - ☐ Private and shared files
- **User administration 👮 and authorization 🔒**
  - ☐ User roles: admin and user
  - ☐ Admin can manage users
- **Address Security 🛡️**
  - ☐ Secure file storage
  - ☐ Secure user authentication
  - ☐ Unauthorized access prevention

Exercise 1 Assignment

- **User authentication 🔑 and file management 📁**
  - ☐ Log in and log out
  - ☐ Upload, download, read and delete files
  - ☐ Private and shared files
- **User administration 👮 and authorization 🔒**
  - ☐ User roles: admin and user
  - ☐ Admin can manage users
- **Address Security 🛡️**
  - ☐ Secure file storage
  - ☐ Secure user authentication
  - ☐ Unauthorized access prevention
- **Address Scalability 🚀 and Test 🧪**
  - ☐ Handle multiple users and files
  - ☐ Test the system performance

Exercise 1 Assignment

- **User authentication 🔑 and file management 📁**
  - ☐ Log in and log out
  - ☐ Upload, download, read and delete files
  - ☐ Private and shared files
- **User administration 👮 and authorization 🔒**
  - ☐ User roles: admin and user
  - ☐ Admin can manage users
- **Address Security 🛡️**
  - ☐ Secure file storage
  - ☐ Secure user authentication
  - ☐ Unauthorized access prevention
- **Address Scalability 🚀 and Test 🧪**
  - ☐ Handle multiple users and files
  - ☐ Test the system performance
- **Production Deployment 🏭 and Cost-Efficiency 💵**

Nextcloud **Built-In** Features

User authentication 🔑 and file management 📁

- ✓ Log in and log out
- ✓ Upload, download, read, delete files
- ✓ Private and shared files

Nextcloud **Built-In** Features

User authentication 🔑 and file management 📁

- ✓ Log in and log out
- ✓ Upload, download, read, delete files
- ✓ Private and shared files

User administration 👮 and authorization 🔒

- ✓ User roles: admin and user
- ✓ Admin can manage users

Nextcloud **Built-In** Features

User authentication 🔑 and file management 📁

- ✓ Log in and log out
- ✓ Upload, download, read, delete files
- ✓ Private and shared files

User administration 👮 and authorization 🔒

- ✓ User roles: admin and user
- ✓ Admin can manage users

Address Security 🛡️

- ✓ Secure file storage
- ✓ Secure user authentication
- ✓ Unauthorized access prevention

Nextcloud **Bonus** Features

- \+ Open source

Nextcloud **Bonus** Features

- \+ Open source
- \+ Multiple plugins

Nextcloud **Bonus** Features

- Open source
- Multiple plugins
- Docker image available

Nextcloud **Bonus** Features

- Open source
- Multiple plugins
- Docker image available
- Helm chart available

Nextcloud **Bonus** Features

- Open source
- Multiple plugins
- Docker image available
- Helm chart available
- Extensive documentation

Nextcloud **Bonus** Features

- Open source
- Multiple plugins
- Docker image available
- Helm chart available
- Extensive documentation
- Compatible with many databases

Nextcloud **Bonus** Features

- \+ Open source
- \+ Multiple plugins
- \+ Docker image available
- \+ Helm chart available
- \+ Extensive documentation
- \+ Compatible with many databases

✓ `Administration settings` → `Security`

Nextcloud **Bonus** Features

- \+ Open source
- \+ Multiple plugins
- \+ Docker image available
- \+ Helm chart available
- \+ Extensive documentation
- \+ Compatible with many databases

- ✓ `Administration settings` → `Security`

- Server-side encryption (SSE)

Nextcloud **Bonus** Features

- \+ Open source
- \+ Multiple plugins
- \+ Docker image available
- \+ Helm chart available
- \+ Extensive documentation
- \+ Compatible with many databases

- ✓ `Administration settings` → `Security`

- Server-side encryption (SSE)
- Password policies

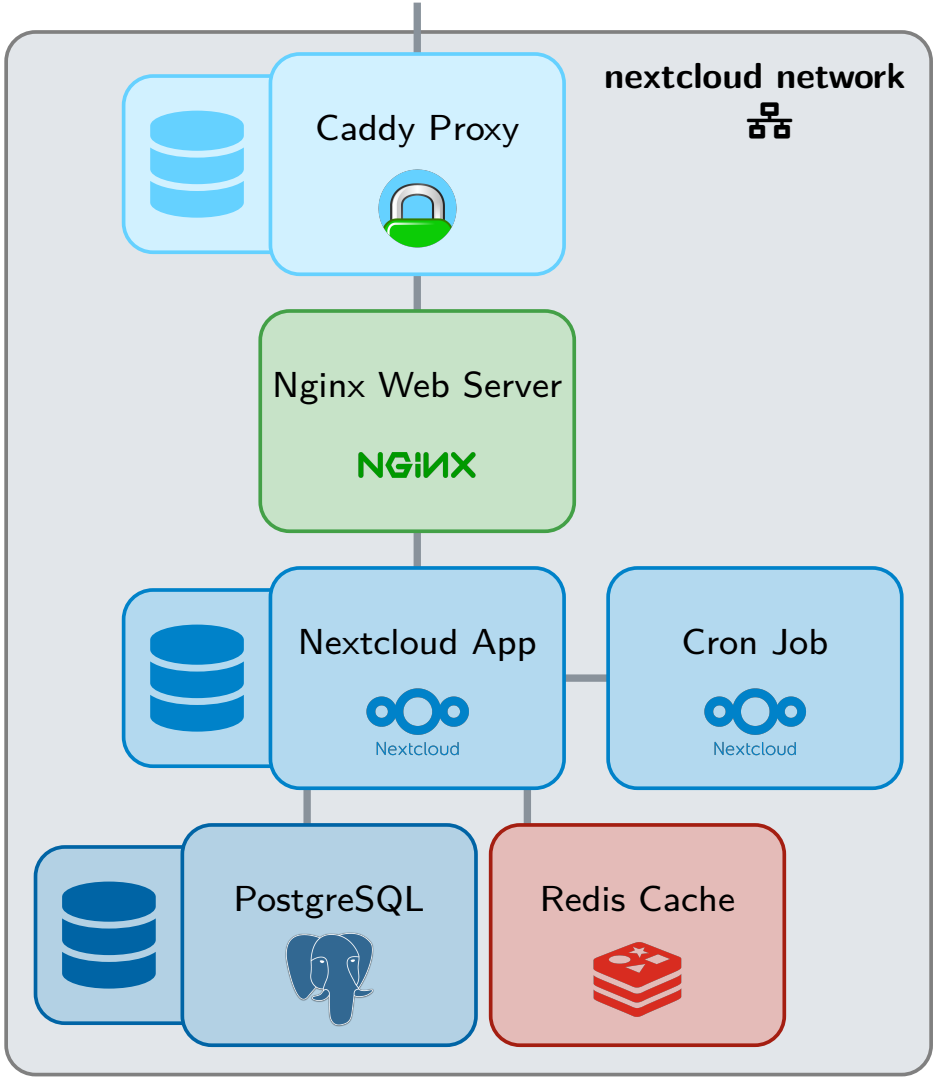
Nextcloud **Bonus** Features

- \+ Open source
- \+ Multiple plugins
- \+ Docker image available
- \+ Helm chart available
- \+ Extensive documentation
- \+ Compatible with many databases

✓ `Administration settings` → `Security`

- Server-side encryption (SSE)
- Password policies
- Two-factor authentication (2FA)

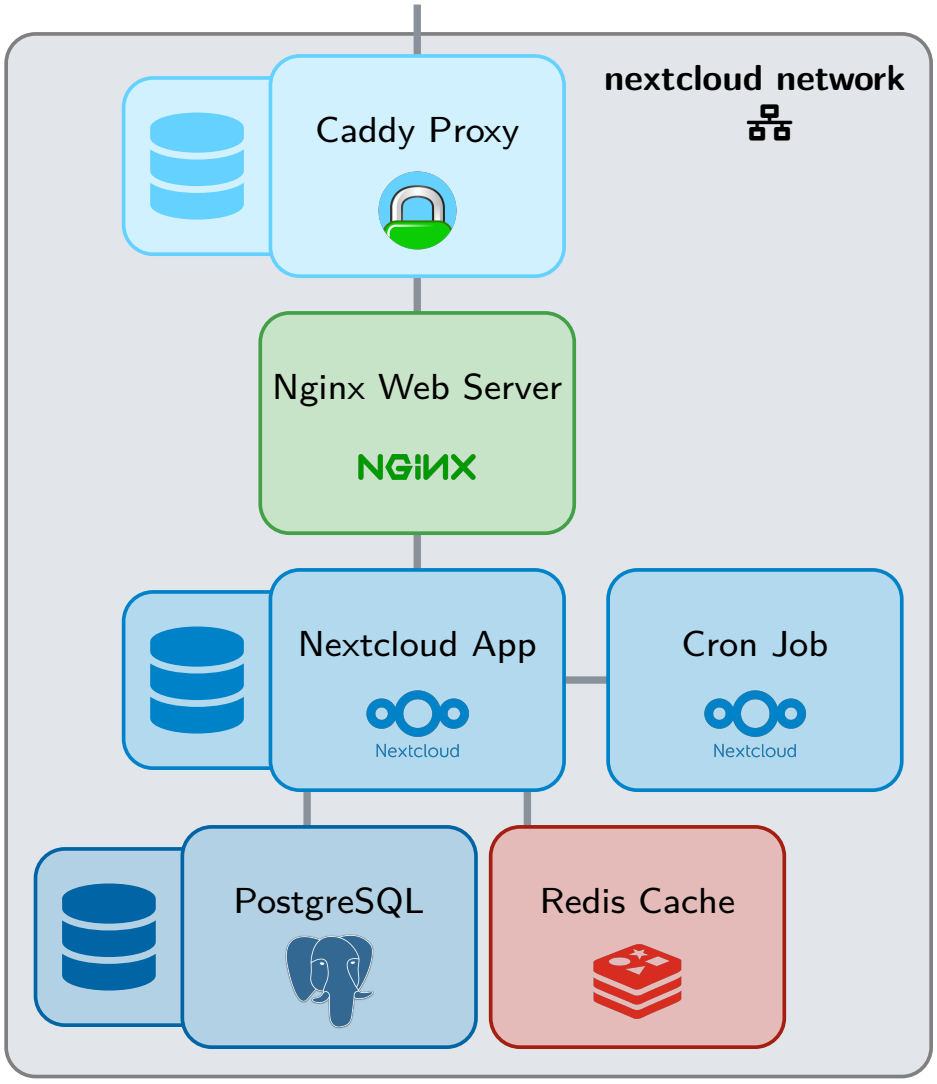
Docker Deployment




```yaml
# Nextcloud app
app:
  image: nextcloud:27.1-fpm
  container_name: app
  restart: always
  networks:
    - nextcloud
  volumes:
    - nextcloud:/var/www/html:z
  environment:
    # ...
    - NEXTCLOUD_ADMIN_USER
    - NEXTCLOUD_ADMIN_PASSWORD
  depends_on:
    - caddy
    - db
    - redis
```

```yaml
# Cron job
cron:
  image: nextcloud:29.0.3-fpm
  container_name: cron
  restart: always
  networks:
    - nextcloud
  volumes:
    - nextcloud:/var/www/html:z
  entrypoint: /cron.sh
  # ...
```

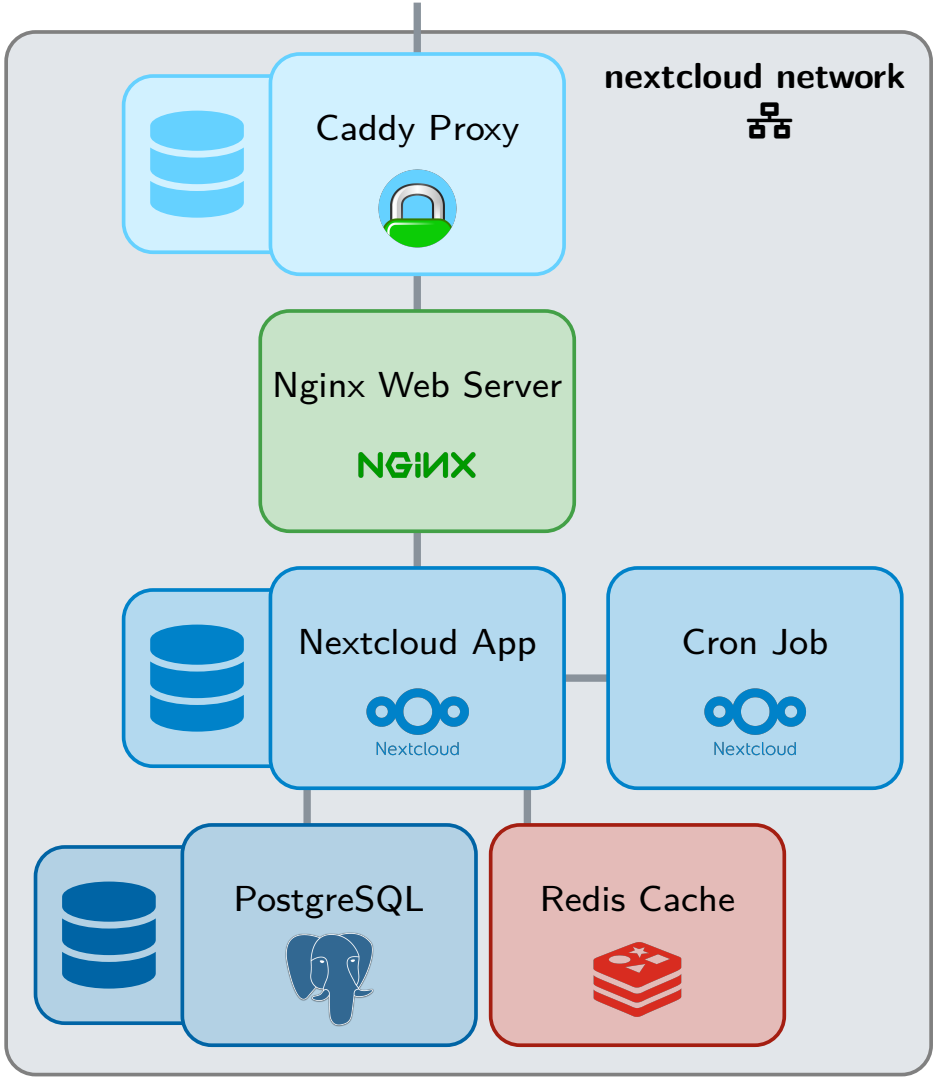
Docker Deployment




```yaml
# PostgreSQL database
db:
  image: postgres:16.3-alpine
  container_name: postgres
  restart: always
  networks:
    - nextcloud
  volumes:
    -
    ↪  db:/var/lib/postgresql/data:Z
  environment:
    - POSTGRES_DB
    - POSTGRES_USER
    - POSTGRES_PASSWORD
    - POSTGRES_HOST

# Redis cache
redis:
  image: redis:7.2.5-alpine
  container_name: redis
  restart: always
  networks:
    - nextcloud
```

Docker Deployment




```yaml
# Nginx web server
web:
  image: nginx:1.27.0-alpine
  container_name: web
  restart: always
  networks:
    - nextcloud
  links:
    - app
  labels:
    - "caddy.reverse_proxy=true"
    # ...
  volumes:
    # ...


# Caddy reverse proxy
caddy:
  image: caddy:2.8.4-alpine
  container_name: caddy
  restart: always
  networks:
    - nextcloud
  ports:
    - 8080:80
    - 443:443
  volumes:
    # ...
```

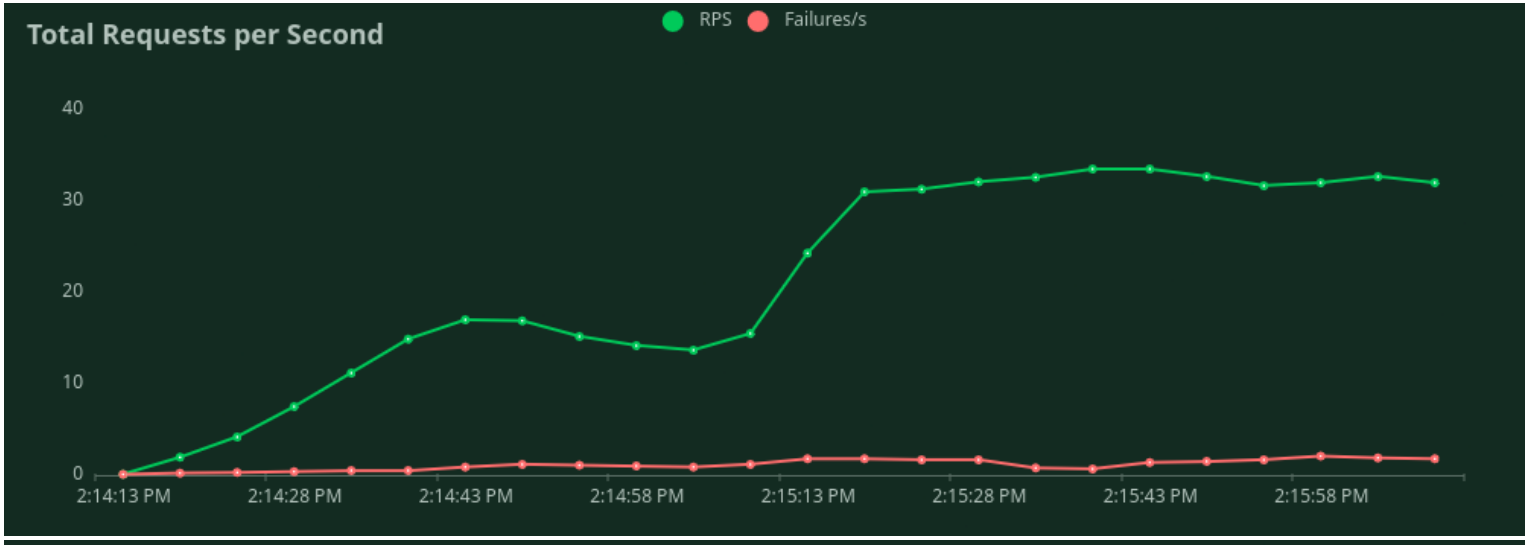
Locust Test

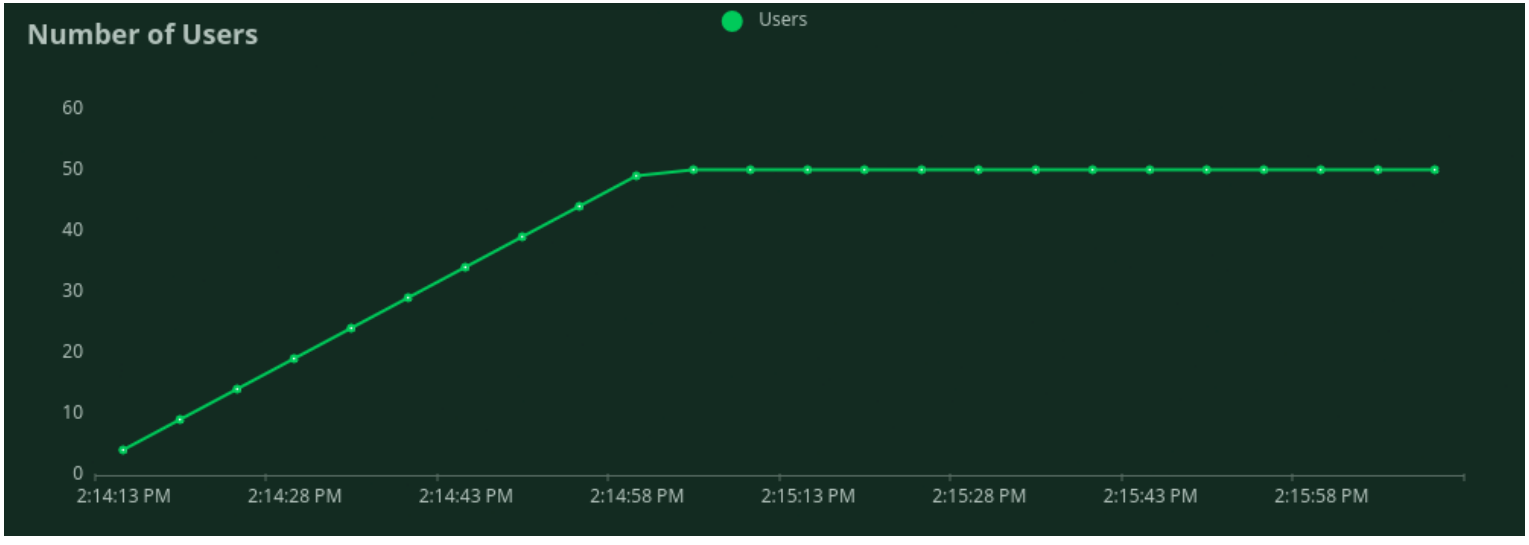
Request	`@task` Ratio	Probability
Read a file	10	32.3%
Download a file	5	16.1%
Upload a 1 KB file	10	32.3%
Upload a 1 MB file	5	16.1%
Upload a 1 GB file	1	3.2%

Table: Different requests and their respective probabilities during the load test.

Locust Test

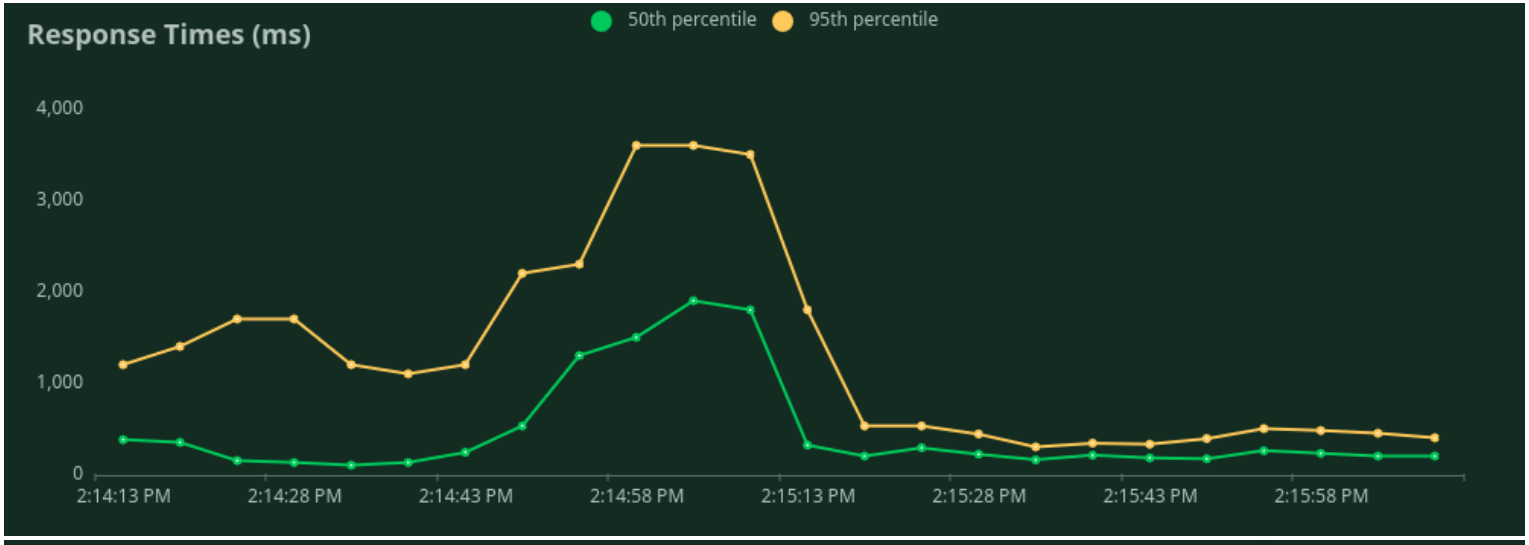
users: 50 | cpu: i7-8565U | ram: 8 GB DDR4

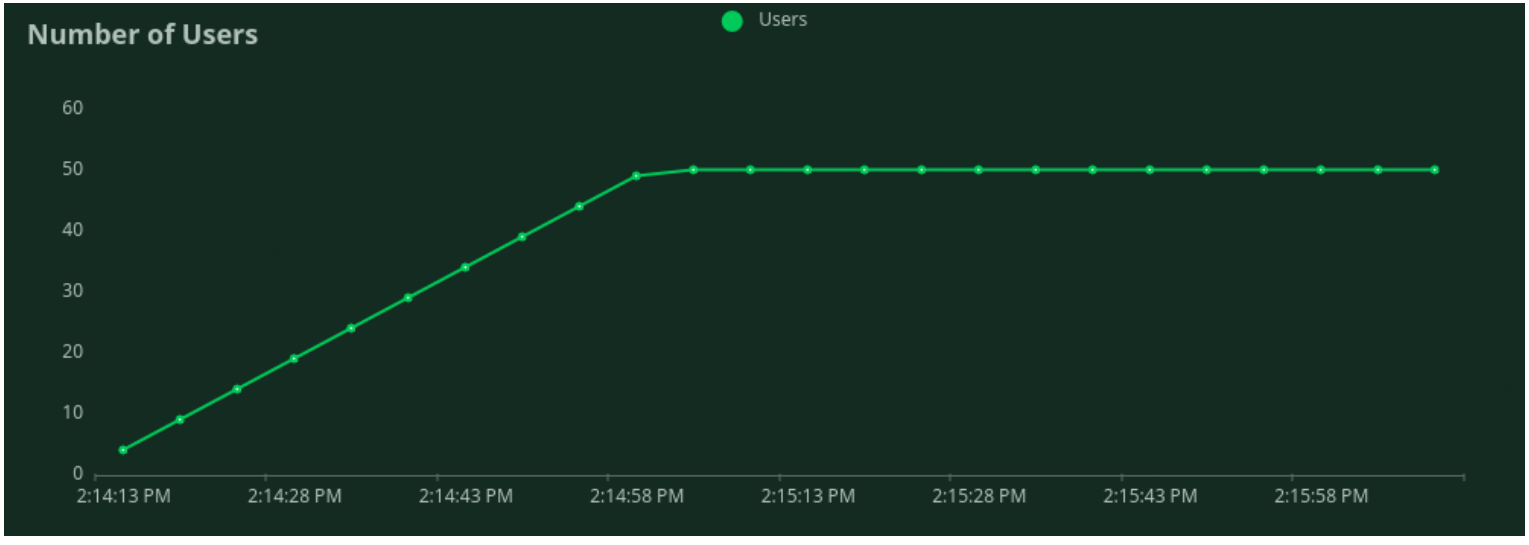




Locust Test

users: 50 | cpu: i7-8565U | ram: 8 GB DDR4





Real-World Deployment

	On-premise	Iaas	Paas	Saas

Real-World Deployment

	On-premise	Iaas	Paas	Saas
Initial Investment	High **$$$**	Medium **$$**	Low **$**	None
Set-Up Time	Long 🕒🕒🕒	Medium 🕒🕒	Short 🕒	None

Real-World Deployment

	On-premise	Iaas	Paas	Saas
Initial Investment	High **$$$**	Medium **$$**	Low **$**	None
Set-Up Time	Long 🕓🕓🕓	Medium 🕓🕓	Short 🕓	None
Hardware Cost	High **$$$**	None	None	None
Maintenance Cost	High **$$$**	Low **$**	None	None
Fees Cost	None	Low **$**	Medium **$$**	High **$$$**

Real-World Deployment

	On-premise	Iaas	Paas	Saas
Initial Investment	High **$$$**	Medium **$$**	Low **$**	None
Set-Up Time	Long 🕒🕒🕒	Medium 🕒🕒	Short 🕒	None
Hardware Cost	High **$$$**	None	None	None
Maintenance Cost	High **$$$**	Low **$**	None	None
Fees Cost	None	Low **$**	Medium **$$**	High **$$$**
Scalability	Limited 🚀	High 🚀🚀🚀	High 🚀🚀🚀	High 🚀🚀🚀
Security	Private	Shared	Your software	Vendor
Control & Privacy	Full	Medium	Low	None

Exercise 2 Assignment

- The cluster must run k8s  ; one node is sufficient
- Pods must have probes to handle miss-behaviors 🌡️
- Volumes must survive pod crash and accidental deletion 🔄
- Service must be accessible via IP or FQDN 🌐
- Databases or third-party services must run in their pod 🗃️

Kubernetes Deployment 

Deployment automated through **provisioning scripts**:

1. Set-up a node using `Vagrant` (*alternative: `Minikube`*)
2.
3.
4.
5.
6.

Kubernetes Deployment 

Deployment automated through **provisioning scripts**:

1. Set-up a node using `Vagrant` (*alternative: `Minikube`*)
2. Install `k8s` (and utilities) through provisioning script
3.
4.
5.
6.

Kubernetes Deployment 

Deployment automated through **provisioning scripts**:

1. Set-up a node using `Vagrant` (*alternative: `Minikube`*)
2. Install `k8s` (and utilities) through provisioning script
3. Deploy `MetalLB` through Helm chart
4.
5.
6.

Kubernetes Deployment 

Deployment automated through **provisioning scripts**:

1. Set-up a node using `Vagrant` (*alternative: `Minikube`*)
2. Install `k8s` (and utilities) through provisioning script
3. Deploy `MetalLB` through Helm chart
4. Deploy `Ingress Nginx` controller through Helm chart
5.
6.

Kubernetes Deployment 

Deployment automated through **provisioning scripts**:

1. Set-up a node using `Vagrant` (*alternative: `Minikube`*)
2. Install `k8s` (and utilities) through provisioning script
3. Deploy `MetalLB` through Helm chart
4. Deploy `Ingress Nginx` controller through Helm chart
5. Apply `PVs`, `PVCs` and `Secrets` for Nextcloud, PostgreSQL and Redis
6.

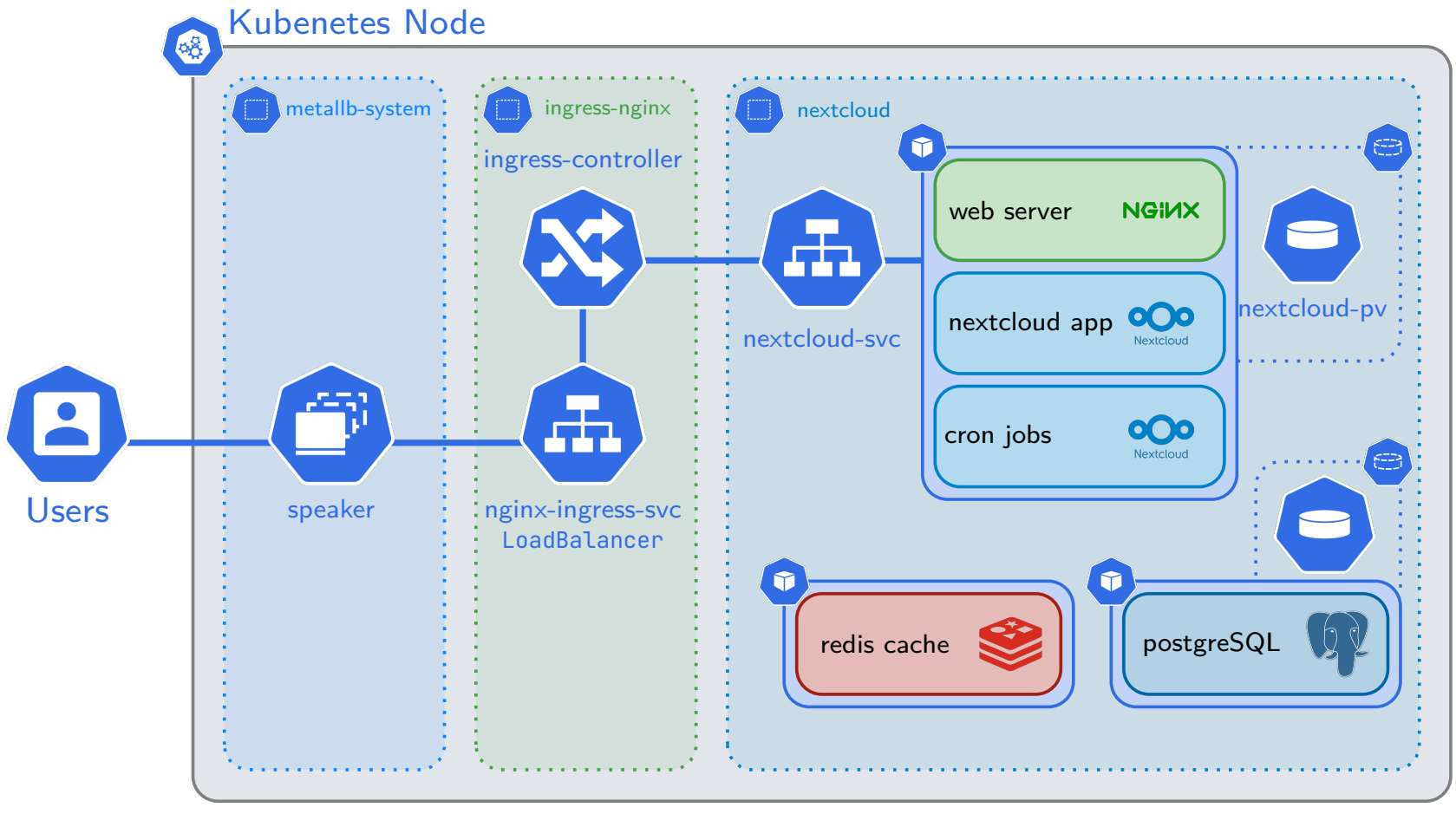
Kubernetes Deployment 

Deployment automated through **provisioning scripts**:

1. Set-up a node using `Vagrant` (*alternative: `Minikube`*)
2. Install `k8s` (and utilities) through provisioning script
3. Deploy `MetalLB` through Helm chart
4. Deploy `Ingress Nginx` controller through Helm chart
5. Apply `PVs`, `PVCs` and `Secrets` for Nextcloud, PostgreSQL and Redis
6. Deploy `Nextcloud` through Helm chart

Kubernetes Deployment

Simplified deployment scheme:



Kubernetes Deployment Features and Advantages

Advantages:

- ✓ **Scalability**: scale up or down pods or replicas
- ✓ **Self-healing**: automatic pod restart in case of failure
- ✓ **Resurces Management**: Horizontal Pod Autoscaler (HPA)
- ✓ **Monitoring**: Startup, Readiness and Liveliness probes
- ✓ **Rolling Updates**: zero update downtime interruption
- ✓ **Secrets Management**: for storing sensitive information
- ✓ **Portability**: Kubernetes is cloud-agnostic
- ✓ **Compatibility**: with many cloud providers
- ✓ **Quick Deployment**: declarative `yaml` files

Disadvantages:

- − **Complexity**: more complex than Docker
- − **Requirements**: 2 GB RAM and 2 CPUs

Exercise 3 Assignment

- The cluster must run `k8s`  ; two node are necessary
- The nodes must talk via either `flannel` or `calico` 🖧
- The `mpi-operator` must be installed
- Create a container with the OSU benchmark 📈
- Estimate the latency between the two nodes ⏱️

Exercise 3 Deployment and Benchmarks

Deployment steps:

1. Set-up 2 nodes using `Vagrant`
2.
3.
4.

Exercise 3 Deployment and Benchmarks

Deployment steps:

1. Set-up 2 nodes using `Vagrant`
2. Install `k8s`, copy `admin.conf` file and add worker node with `kubeadm join`
3.
4.

Exercise 3 Deployment and Benchmarks

Deployment steps:

1. Set-up 2 nodes using `Vagrant`
2. Install `k8s`, copy `admin.conf` file and add worker node with `kubeadm join`
3. Install `flannel` and set-up flannel network through Helm
4.

Exercise 3 Deployment and Benchmarks

Deployment steps:

1. Set-up 2 nodes using `Vagrant`
2. Install `k8s`, copy `admin.conf` file and add worker node with `kubeadm join`
3. Install `flannel` and set-up flannel network through Helm
4. Install `mpi-operator` with specialized containers deployment and create a `mpi-job` (`yaml` file)

Exercise 3 Deployment and Benchmarks

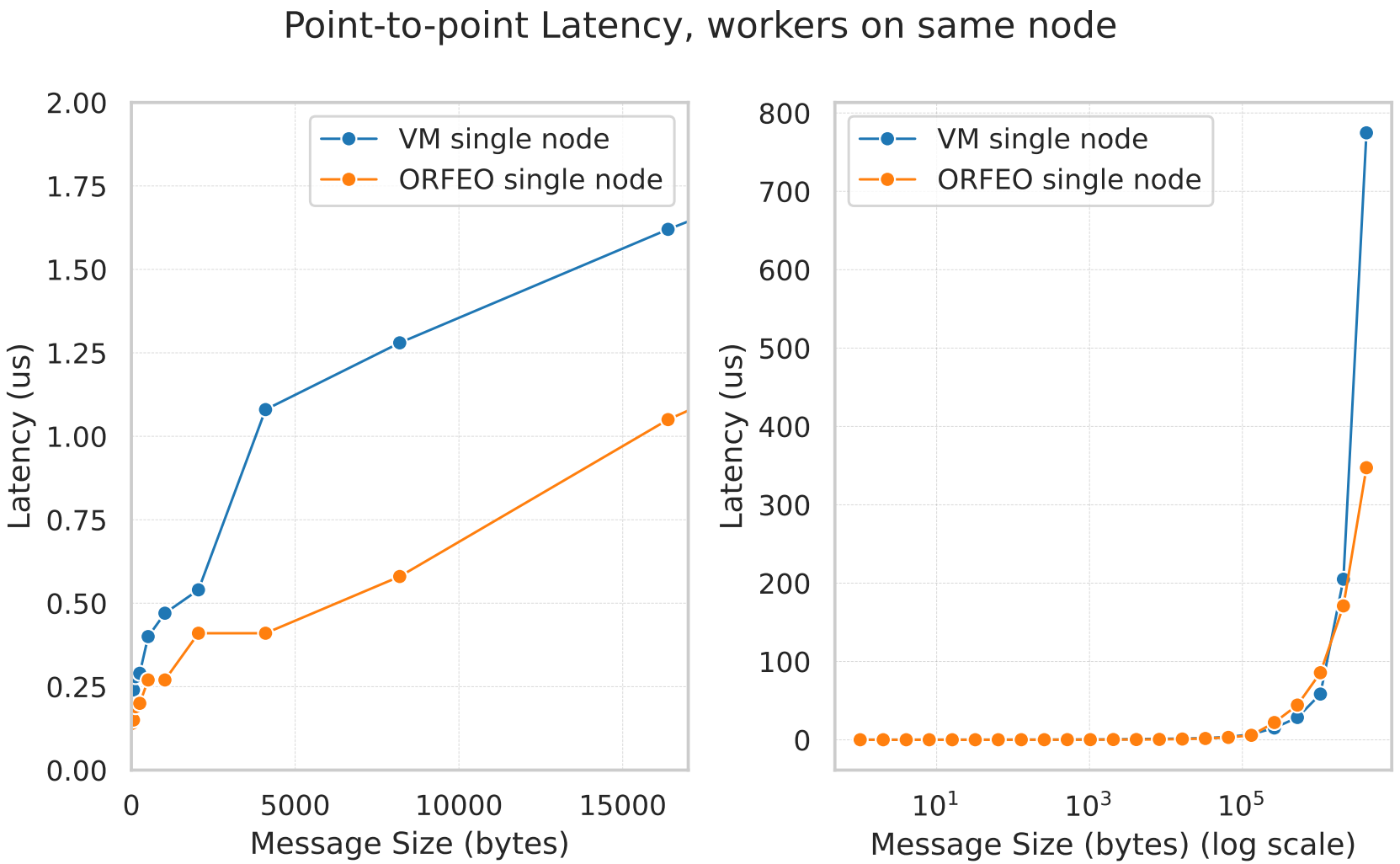
Deployment steps:

1. Set-up 2 nodes using `Vagrant`
2. Install `k8s`, copy `admin.conf` file and add worker node with `kubeadm join`
3. Install `flannel` and set-up flannel network through Helm
4. Install `mpi-operator` with specialized containers deployment and create a `mpi-job` (`yaml` file)

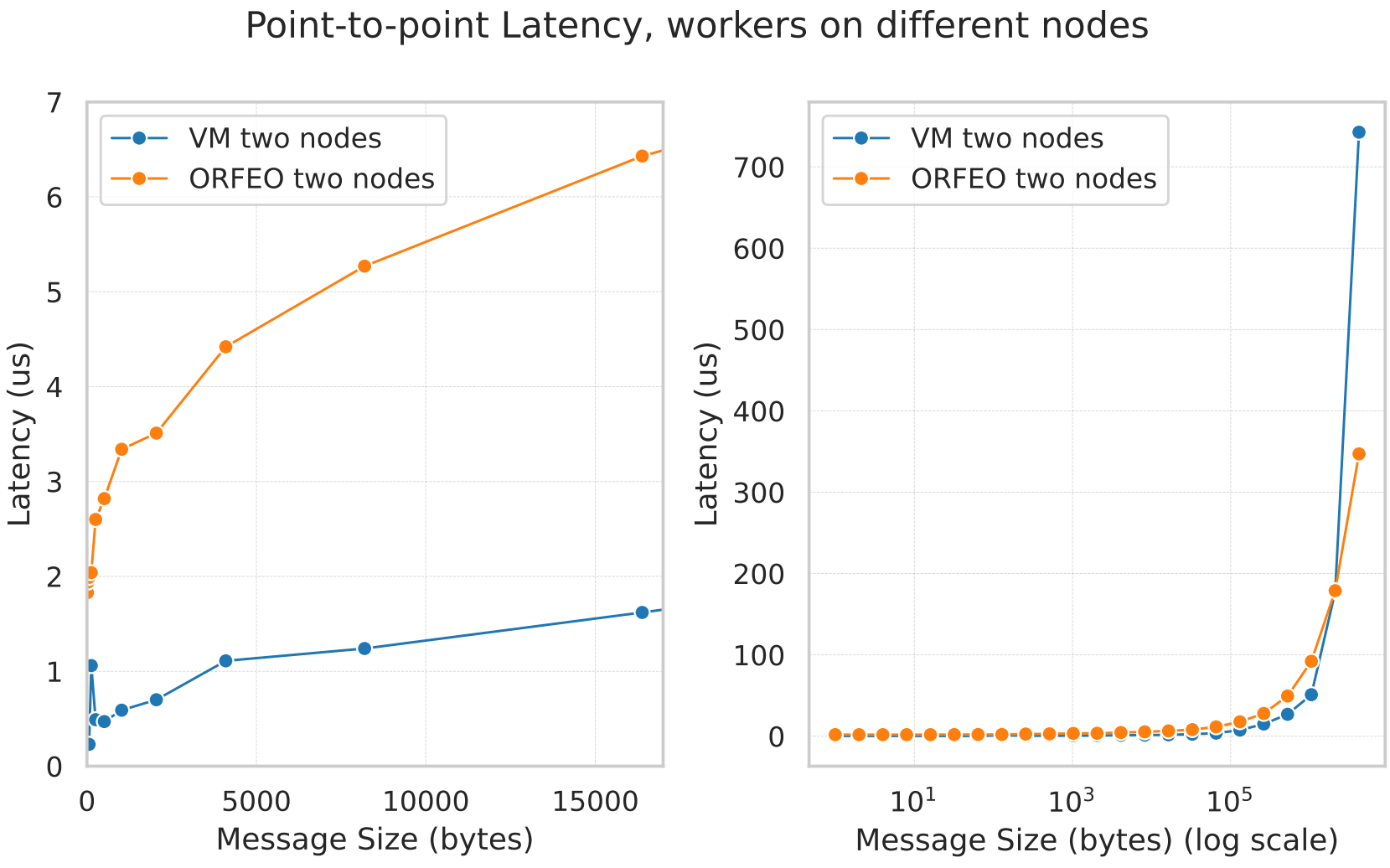
Conducted benchmarks:

- Point-to-point latency test
- Broadcast latency test

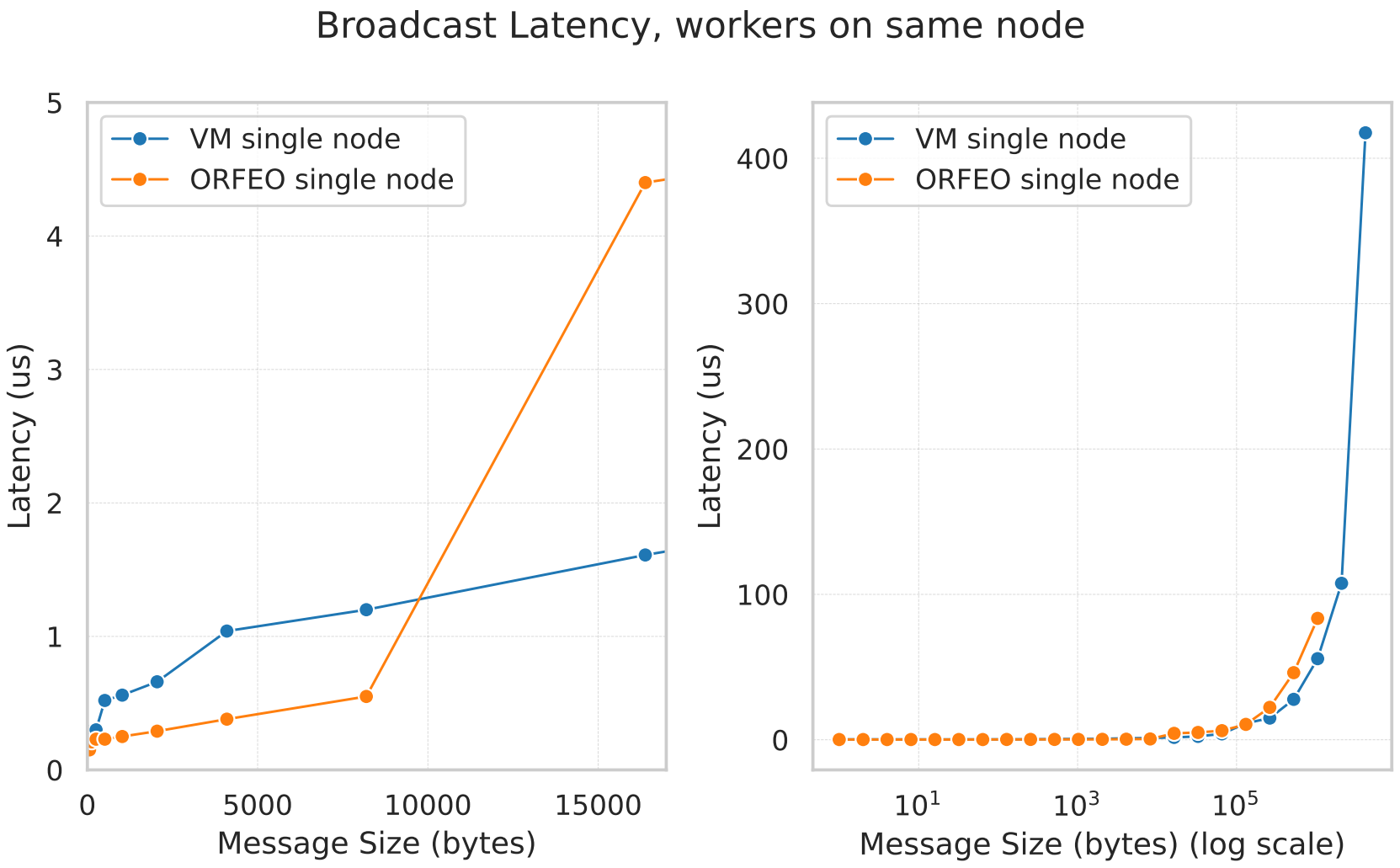
OSU Benchmark: point-to-point latency test



OSU Benchmark: point-to-point latency test



OSU Benchmark: Broadcast latency test



OSU Benchmark: Broadcast latency test