computers

Article

Educational Resource Private Cloud Platform Based on OpenStack

Linchang Zhao [1,*], Guoqing Hu [2,*] and Yongchi Xu [1]

[1] School of Computer Science, Guiyang University, Guiyang 550005, China; yongchixu@126.com
[2] PKU-HKUST Shenzhen-Hong Kong Institution, Shenzhen Institute of Peking University, Shenzhen 518057, China
* Correspondence: anilue@alu.cqu.edu.cn (L.Z.); huking@pku.edu.cn (G.H.); Tel.: +86-13364059762 (L.Z.); +86-18585876451 (G.H.)

Abstract: With the rapid development of the education industry and the expansion of university enrollment scale, it is difficult for the original teaching resource operation and maintenance management mode and utilization efficiency to meet the demands of teachers and students for high-quality teaching resources. OpenStack and Ceph technologies provide a new solution for optimizing the utilization and management of educational resources. The educational resource private cloud platform built by them can achieve the unified management and self-service use of the computing resources, storage resources, and network resources required for student learning and teacher instruction. It meets the flexible and efficient use requirements of high-quality teaching resources for student learning and teacher instruction, reduces the construction cost of informationization investment in universities, and improves the efficiency of teaching resource utilization.

Keywords: OpenStack and Ceph; virtualized resources; private cloud platform

Citation: Zhao, L.; Hu, G.; Xu, Y. Educational Resource Private Cloud Platform Based on OpenStack. *Computers* **2024**, *13*, 241. https://doi.org/10.3390/computers13090241

Academic Editor: Rao Mikkilineni

Received: 25 August 2024
Revised: 8 September 2024
Accepted: 10 September 2024
Published: 23 September 2024

1. Introduction

As education expands, focus shifts from teacher-centric to student-centric resource demands. Growing student numbers and broader education needs highlight the need for innovative learning environments with tailored resources [1]. However, open educational resources (OERs) face challenges in monitoring progress, with educators hesitant due to control concerns [2]. OERs may lack accuracy, completeness, and customization options [3].

OpenStack and Ceph offer open-source solutions for large-scale virtualization, supporting petabytes of data, unlimited scale, and configurable networking [4]. Ideal for education and government, they reduce costs and optimize resource utilization. Cloud providers offer software-as-a-service (SaaS), platform-as-a-service (PaaS), and infrastructure-as-a-service (IaaS) services [5]. Bhatia et al. [6] have provided a design of a private cloud for higher education and proof-of-concept implementation methodology for the OpenStack platform and analyzed the advantages and disadvantages of a private cloud.

Recent experimental research on the application of the open-source infrastructure service platform OpenStack underscores that its key technology aspects provide a solid foundation for advancing modern technology [7]. Cloud management platforms (CMPs) play a pivotal role in sustaining both private and public cloud computing by facilitating the management, provisioning, and monitoring of resources and their utilization [8].

Cloud computing enables low-cost access to shared resources. Load balancing optimizes resource use, preventing overload/underload due to operating system updates, task times, server failures, etc., [9]. OpenStack uses orchestration for cloud service deployment and management [10]. Aregbesola et al. [11] compared cloudStack, OpenStack, and Eucalyptus for business and research, assessing their strengths, limitations, and viability.

IaaS offers flexibility and control over networking resources [12]. Private clouds blend public cloud benefits with enhanced control and customization [13]. HA systems ensure resilience through redundant nodes and clusters [14]. OpenStack, a cloud OS, manages computing, storage, and network resources efficiently [15]. Lima et al. analyze OpenStack's growth and market impact, guided by the OpenStack Foundation [16]. Zhang et al. [17] have developed a distributed block-level storage system called ORTHRUS, which is integrated into this IaaS system, and deployed a Ganglia monitoring system in the IaaS system to monitor all the physical node VM instances.

The objective is to establish a private cloud model with OpenStack for IaaS, PaaS, and SaaS [18]. As an open-source cloud platform, OpenStack enables users to deploy service models. Integrating OpenStack with Ceph distributed storage creates a flexible educational resource private cloud for the unified management and self-service utilization of computing, storage, learning, and network resources [19]. Bonner et al. [20] introduced the deployment methods and configurations for OpenStack along with the security provisions that were taken to deliver computer hardware. The rationale behind the provisions of virtual hardware and OS configurations have been defined in great detail, supported by examples. This study constructs a teaching-focused private cloud management platform to improve resource management efficiency and standards. The key contributions of this paper encompass the following:

(1) This paper presents the construction scheme of a campus teaching private cloud platform.
(2) This paper introduces the construction process of a campus teaching private cloud platform in detail.
(3) It provides a solution for the management and full utilization of teaching resources.

The rest of this paper is organized as follows. Section 2 introduces the related work. Section 3 explains combining OpenStack with Ceph. Section 4 explains the design of the educational resource private cloud platform. Section 4 shows the implementation of the educational resource private cloud platform. The implementation of HA for the OpenStack educational resource private cloud platform is introduced in Section 5. The test and analysis of the private cloud platform for educational resources is shown in Section 6. Finally, Section 7 presents the conclusions.

2. Related Work

2.1. OpenStack

OpenStack, a NASA–Rackspace collaboration, is an Apache-licensed open-source cloud computing software. It offers a full suite of cloud platform solutions for computing, storage, networking, and developer APIs [21]. Enabling seamless app creation, deployment, and management, OpenStack provides stable, flexible, reliable, and scalable cloud services for diverse cloud businesses.

The OpenStack community embraces open development, welcoming all to contribute code, functions, tests, etc. Since "Austin", releases follow a six-month cycle, named alphabetically [22]. OpenStack runs on diverse hardware and operating systems, supports various virtualizations, and is popular in production for public, private, and hybrid clouds.

OpenStack's core is resource virtualization, centered on VMs. Its seven key modules are as follows: Nova (compute), Neutron (networking), Cinder (block storage), Swift (object storage), Keystone (auth/authz), Glance (image mgmt), and Horizon (UI) [23]. Together, they offer VM computing, networking, and storage resources.

2.2. Ceph

Ceph, a popular distributed storage system, offers scalable, high-performance, and available storage. It integrates resources across nodes, providing block, file, and object storage. Data are distributed and replicated, supporting POSIX and RESTful APIs.

Ceph ensures data reliability with multi-replica storage [24]. It automatically stores replicas to prevent loss from disk failures, crashes, outages, etc. Ceph clusters have multiple nodes for storage, computing, and networking, ensuring continuous data reliability.

Ceph comprises OSDs, Monitors, and MDSs for functionality. OSDs manage cluster data, handling snapshots, replication, recovery, and migration, while monitoring OSD processes. Monitors maintain cluster state charts, including OSD, placement group, and

CRUSH maps. MDSs handle metadata computation, caching, and sync for consistency, ensuring Ceph's scalability, performance, and reliability.

3. OpenStack with Ceph

OpenStack and Ceph deliver high-performance, reliable, scalable private clouds, catering to the flexible and efficient use of teaching resources [25]. Ceph, a distributed storage platform, balances loads with CRUSH for low-latency, high-throughput access. OpenStack manages and schedules computing resources optimally [26]. Ceph, as OpenStack's storage backend, (1) cuts costs by avoiding expensive storage; (2) offers diverse storage types; and (3) provides persistent storage services.

OpenStack and Ceph ensure high reliability with data redundancy, recovery, fault isolation, and self-healing [27]. Ceph uses replication/erasure coding for data reliability and auto-recovery [28]. OpenStack has robust fault detection and isolation, with automated scheduling and load balancing for stable operations [29]. OpenStack utilizes Ceph's distributed storage for data archiving and sharing, with concurrent reads. Ceph setup initially needs firewall off for communication, supporting block, file, and object storage.

The scalability and elasticity of OpenStack with Ceph are bolstered by their storage and compute expansion features. Ceph's distributed design allows for seamless storage scaling; new nodes can be added to the cluster, which Ceph integrates into storage pools for linear capacity increases [30]. OpenStack's nova component gathers data via the nova-scheduler, records them in the database, and then dispatches them to a message queue, which reassigns tasks to the nova-scheduler. OpenStack dynamically adds/removes compute nodes to adapt to changing demands [31]. Adding nodes boosts power, while removing excess ones conserves resources, making it ideal for private cloud education platforms with Ceph.

4. Design of Educational Resource Private Cloud Platform

Prior to designing the private cloud platform, we surveyed educational institutions, teachers, and students on cloud resource needs and usage. Analysis of the public ed cloud DB literature revealed data inaccuracies, incompleteness, and privacy/security weaknesses [32]. Based on the identified research gaps, this paper clarified the research direction and focus, proposed corresponding research strategies, and determined the database design and implementation plan.

4.1. Functional Architecture Design

The educational resource private cloud platform boasts a three-tier functional architecture: Infrastructure, Resource Management, and Resource Application. This architecture underpins cloud-based services like computing, storage, and networking, accommodating devices like x86 servers, mainframes, and storage devices. It fosters seamless management and self-service access to educational resources, as depicted in Figure 1.

Infrastructure Layer: Serving as the cornerstone of the cloud computing platform, this tier encompasses servers, storage, networking, and security devices. Leveraging x86 servers, it enables resource migration via advanced technologies. Virtualization underpins the cloud infrastructure, enabling dynamic resource allocation and streamlined educational resource management [33].

Resource Management Layer: The cloud computing platform's core tier encompasses resource partitioning, high availability, elastic allocation, load balancing, and security. With compute, storage, and network virtualization, it securely supports upper-level services, providing tailored, efficient, and reliable cloud resources [34].

Resource Application Layer: The top layer of the cloud platform offers users access via various OSs on PCs/mobiles. Users, like students and teachers, can request tailored resources like computing, storage, and education. They can also customize servers, OS, experimental platforms, and configure computing, storage, and network resources [35].

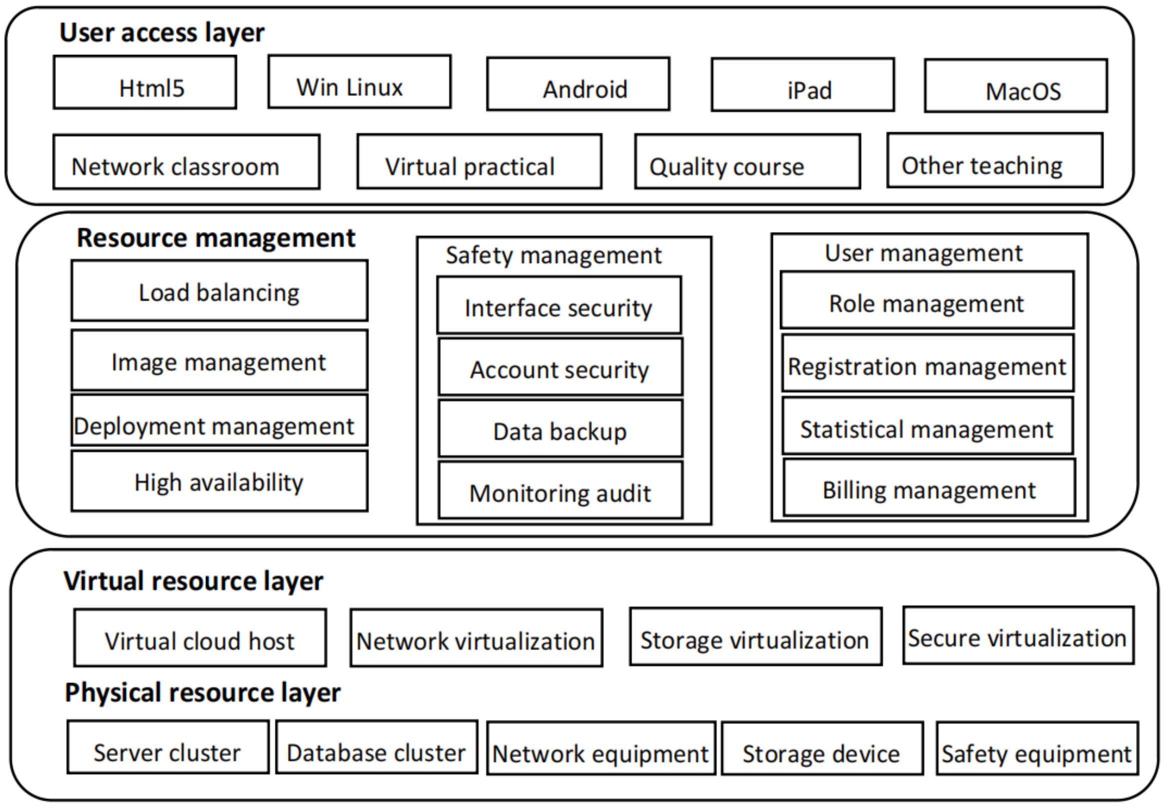


Figure 1. Schematic diagram of the Ceph distributed database structure.

4.2. Deployment Architecture Design

The cloud platform built with OpenStack combined with Ceph adheres to the principle of "centralized control and division of labor". Based on the high degree of loose coupling and distributed characteristics among the modules, the platform adopts a multi-node deployment design. The server configuration information for each node is as shown in Table 1.

Table 1. Server configuration.

Nodes	Configuration
Proxy Nodes	1 × CPU Intel i9-13900K with 24 cores, 4 × 32 GB RAM, 2 × 2 TB SAS disks, 2 × 2-port Gigabit Ethernet network cards
Control Nodes	2 × CPUs Intel i9-13900K with 24 cores each, 5 × 32 GB RAM, 2 × 2 TB SAS disks, 2 × 2-port Gigabit Ethernet network cards
Network Nodes	2 × CPUs Intel i9-13900K with 24 cores each, 5 × 32 GB RAM, 2 × 2 TB SAS disks, 2 × 2-port Gigabit Ethernet network cards
Compute Nodes	2 CPUs of Intel i9-13900K with 24 cores each, 5 × 32 GB RAM, 2 × 2 TB SAS disks, and 2 × 2-port Gigabit Ethernet network cards.
Storage Nodes	2 × Intel i9-13900K CPUs with 24 cores each, 5 × 32 GB RAM, 2 × 2 TB SAS disks, and 2 × 2-port Gigabit Ethernet network cards

The platform environment utilizes thirteen Great Wall servers, with two configured as proxy nodes, two as highly available control nodes, three as highly available network nodes, three as compute nodes, and three as Ceph distributed storage nodes. Each server is equipped with 2 Intel i9-13900K CPUs, each CPU featuring 24 cores, 32 GB of RAM, 2 TB SAS hard drives, and dual-port Gigabit Ethernet network cards for Ethernet connectivity.

Additionally, there are auxiliary devices such as routers, switches, and external networks. The detailed deployment architecture is illustrated in Figure 2.

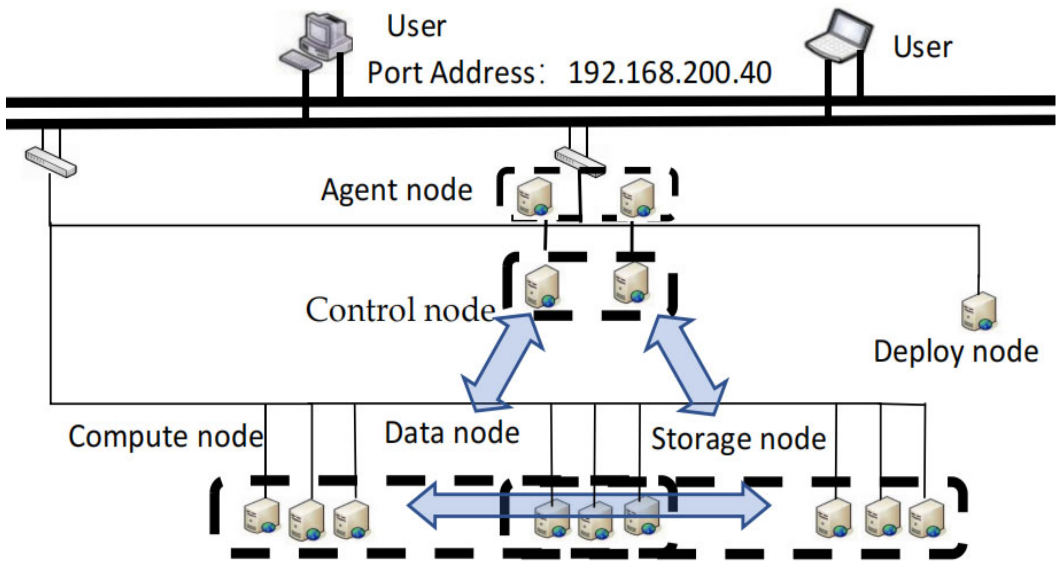


Figure 2. Schematic diagram of the deployment architecture.

Proxy Nodes: For cloud platforms based on OpenStack, communication is mostly performed through APIs using the TCP protocol to control traffic. Our private cloud platform is no exception, utilizing Haproxy for traffic monitoring, distribution, and restriction. In the haproxy.cfg file, traffic distribution protocols are configured. Haproxy provides the Leastconn algorithm, which directs new requests to the server with the least pressure, maintaining long-lasting sessions without automatic interruption and without connection limits. Additionally, Keepalived's Virtual IP (VIP) technology safeguards against Haproxy's single-node failure. During operation, the VIP constantly monitors node heartbeats. If a node remains active, the VIP stays put. However, upon detecting a node failure, the VIP seamlessly migrates to a backup node, maintaining proxy functionality. This prevents downtime and ensures platform stability and continuous availability [36].

Control Nodes: The platform boasts a high-availability architecture with two control nodes, each hosting Keystone, Glance, Placement, Nova, Neutron, Horizon, and Cinder. Deployment starts with the primary Keystone node and its keystone.conf configuration, followed by a backup node with synchronized keystone.conf. Haproxy proxies control both nodes, completing the high-availability setup. Glance, Placement, Horizon, and Cinder also have primary and backup nodes, with Haproxy managing traffic between them. Nova and Neutron focus on control node services, with Haproxy overseeing traffic control and monitoring. This custom-built cloud platform guarantees high availability, scalability, and stability, catering to the educational needs of students and teachers with premium resource services [37].

Network Nodes: The Neutron component serves as the networking element within the OpenStack cloud computing platform. It is responsible for managing and connecting networks between virtual machines (VMs) and other computing resources. Neutron, with its extensive API and plugin system, supports diverse network topologies and services, facilitating the management, configuration, and scheduling of both virtual and physical networks. It covers IP allocation, connectivity, security policies, and more, while provisioning isolated virtual networks for tenants, enforcing network security and resource separation through isolation policies. Neutron's plugin mechanism seamlessly integrates with hardware/virtualization platforms like OVS and Linux Bridge, ensuring flexibility and scalability. Its L2/L3 agents and plugins enable advanced networking technologies (GRE tunnels, VLANs, VxLANs), connecting VMs to the physical WAN and facilitating VM-to-VM communication [38].

Compute Nodes: The private cloud platform leverages multiple x86 servers for computational tasks, catering to students' learning and teaching needs. It ensures the scalability and expandability of compute nodes, maintaining business continuity during upgrades, without disrupting the platform. Nova and Neutron collaborate on three servers to virtualize CPU, memory, storage, and network resources, enabling the deployment and management of multiple cloud hosts. The efficient operation of the cloud platform relies on Nova and Neutron's virtualization, along with Nova-compute and Neutron-openvswitch-agent services. Yet, for sustained and stable performance, the private cloud platform necessitates thorough consideration of technical support and parameter optimization across networking, storage, and security domains [39]. The deployment of compute node servers is illustrated in Figure 3.

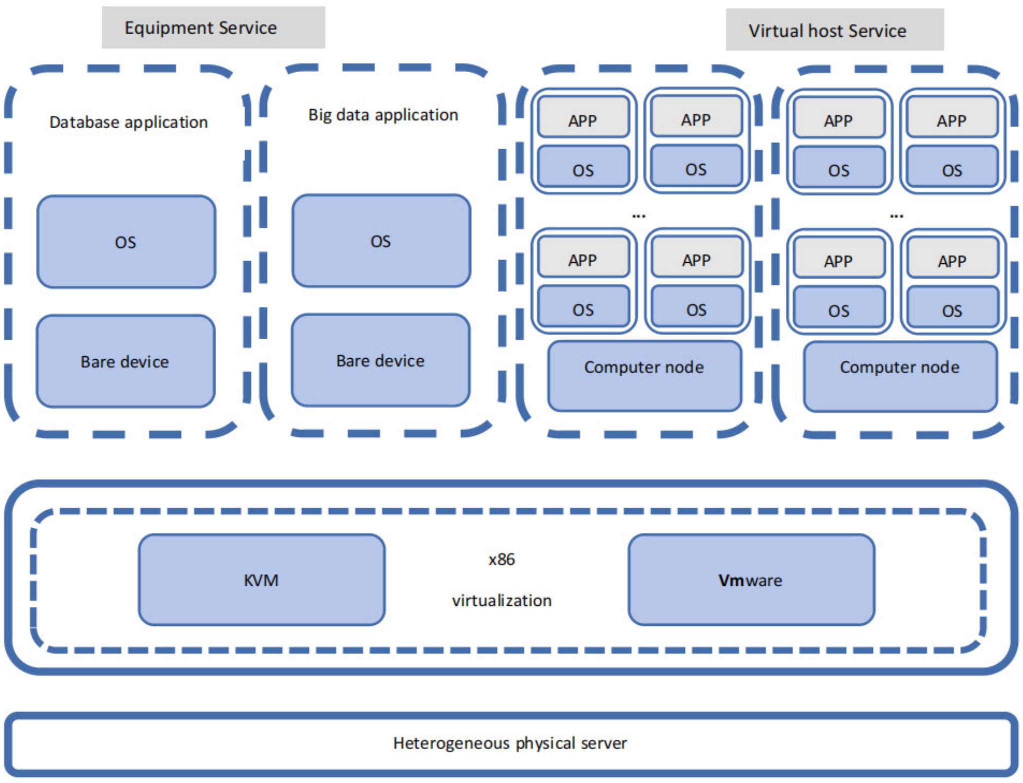


Figure 3. Schematic diagram of compute node server deployment.

Data Nodes: The cloud platform's data interactions are orchestrated by three key components: MariaDB cluster, RabbitMQ cluster, and Memcache cluster. MariaDB, a relational database server, houses libraries and tables vital to platform components. RabbitMQ, leveraging its clustering capability, forms a three-server cluster for real-time message queuing and synchronization. The Memcache cluster enhances platform efficiency by caching temporary data, minimizing database hits and load. For high availability and real-time data sync, we opted for the Galera Cluster's multi-master mode, enabling concurrent service provision across nodes and optimizing resource utilization over traditional master–slave setups [40]. The deployment of the data servers is illustrated in Figure 4.

Storage Nodes: Utilizing Ceph distributed storage technology, we have constructed a distributed storage resource pool that aggregates the hard drives of three servers, forming a large storage pool. Our privately built cloud platform provides two types of storage: block storage and shared storage. Block storage is used to store data blocks, while shared storage is used for file sharing. To enhance reliability, we have deployed three Monitor monitors, two Manager cluster managers, and three Metadata Server metadata servers in this solution [41]. The planning and design of these servers adopt a multi-node and fault-tolerant approach, ensuring that even if one server fails or one-third of the total number of hard drives become dysfunctional, it will not affect the normal operation of the entire distributed storage cluster. This solution can also provide storage services for various types of data. The deployment of storage node servers is illustrated in Figure 5.

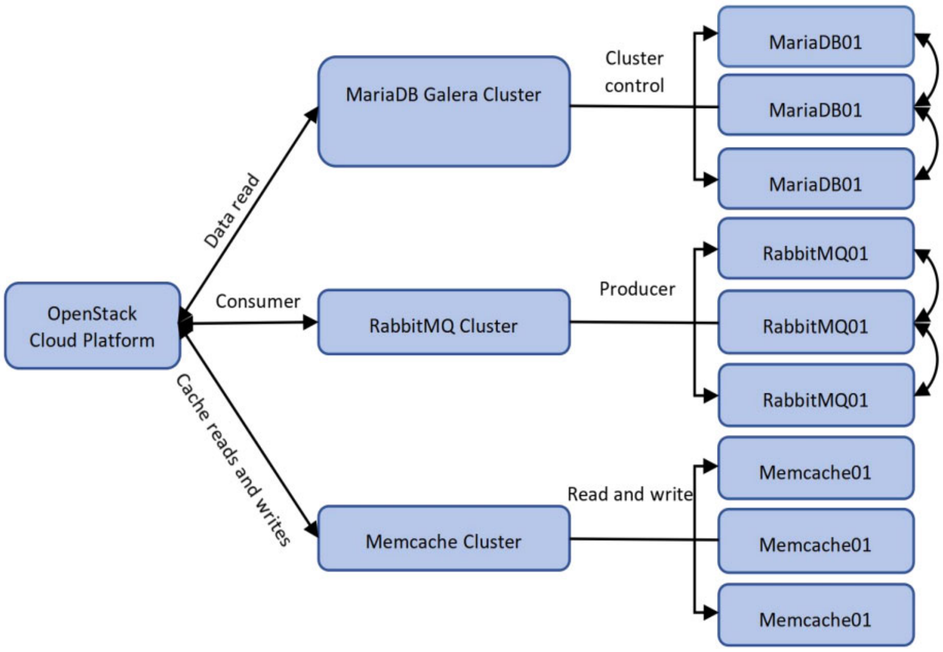


Figure 4. Schematic diagram of data server deployment.

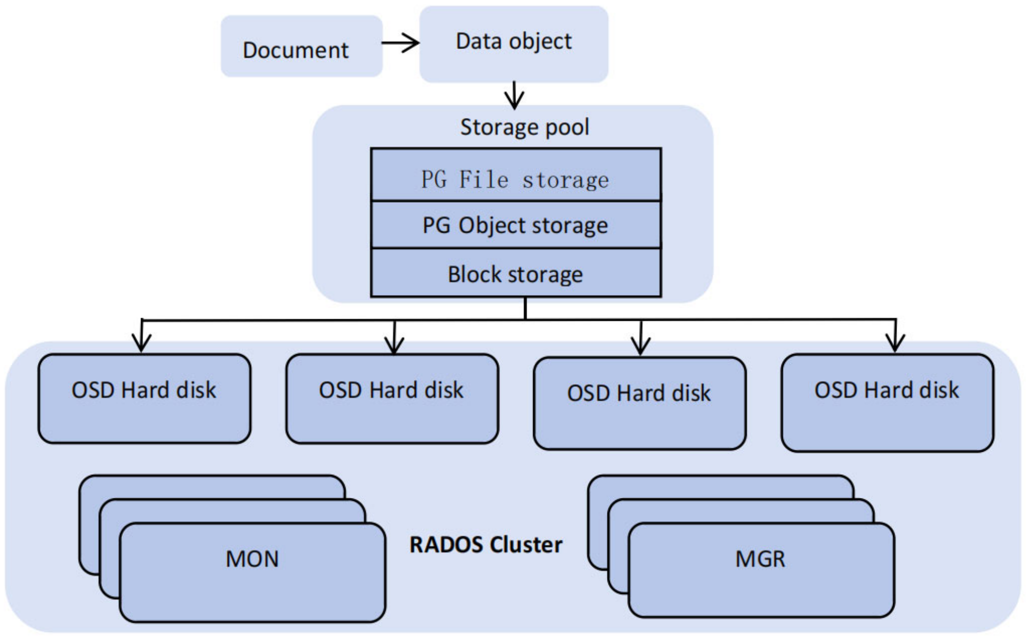


Figure 5. Schematic diagram of storage node server.

5. Implementation of HA for OpenStack Educational Resource Private Cloud Platform

Currently, the data accuracy and integrity of open-source public educational cloud resource databases are insufficient, user privacy protection and data security are relatively weak, and there is insufficient support for highly customizable and directly controllable resource allocation, resulting in the inefficient utilization of teaching resources that cannot meet the needs of high-quality education development. To address these issues, this paper proposes the construction of a teaching resource private cloud platform based on the open-source OpenStack. This platform not only supports deep customization to meet specific business needs and operational processes but also allows for complete control over its data and applications, enabling the configuration of security measures based on specific business security standards. By incorporating a modular architecture design, independent upgrades and expansions of individual components can be achieved. Additionally, various virtualization technologies allow users to further optimize private cloud performance and costs based on their requirements and preferences.

5.1. Deployment of Basic Environment

This paper adopts the stable versions from the official websites of OpenStack and Ceph to construct a private cloud platform for teaching resources. The primary focus is on deploying and configuring key components such as proxy nodes, control nodes, network nodes, data nodes, compute nodes, and storage nodes. The required hardware and software resources are as follows:

Hardware Resources: Utilize 13 physical servers to set up the required 13 nodes for the platform. The total resource allocation across all servers includes 128 CPU cores, 768 GB of RAM, 20 TB of hard disk space, and 10 Gbps Network Interface Cards (NICs). Each node is equipped with two network cards, and two Layer 3 switches along with a router are used for communication configuration.

Software Resources: Each server is installed with Ubuntu 22.04 as the system environment. The private cloud platform for teaching resources, built using two control nodes and three compute nodes, uses OpenStack's Yoga release software repository. The three storage nodes are set up with Ceph Quincy release, and the three data nodes are used to deploy services such as MariaDB, Memcached, and RabbitMQ. After the system environment is deployed on each server, the basic server environment is initialized, such as hostname and network card segment configuration.

5.2. HA Implementation for Proxy Nodes

Proxy servers are a common network architecture that can help manage and secure server access. Implementing a proxy server using HAProxy and Keepalived is a simple yet reliable method to enhance server availability and security.

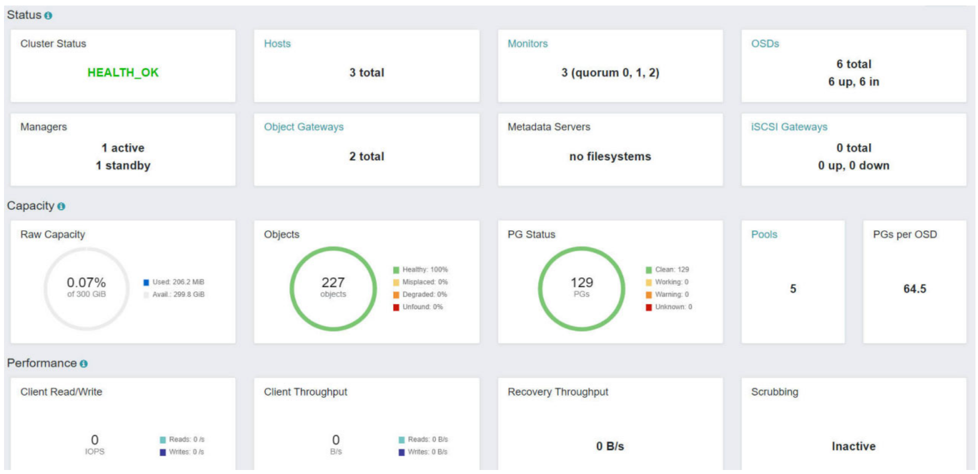
Firstly, it is necessary to install and configure HAProxy and Keepalived. When configuring HAProxy, one can achieve a proxy server by adding listeners for OpenStack service ports and configuring relevant options and servers. For instance, it can listen to HTTP services (port 80) and HTTPS services (port 443), while configuring the IP addresses and port numbers of backend OpenStack servers.

Using Keepalived, one can configure high availability for the HAProxy instances, implementing a master–backup mechanism. This ensures that if the primary instance fails, the backup instance can automatically take over the service, maintaining the continuous availability of OpenStack services.

Finally, testing is required to ensure that the proxy server correctly forwards requests to the backend OpenStack servers. By accessing the IP addresses and port numbers of OpenStack services, the normal operation of the proxy server can be tested. The implementation steps are as follows:

Update configuration /etc/nova/nova.conf; /etc/neutron/plugins/ml2/ml2_conf.ini;

Open vSwitch, configuring OVS, ovs-vsctl add-br br-eth1; ovs-vsctl add-port br-eth1 eth1; restart service, systemctl restart nova-compute; systemctl restart neutron-openvswitch-agent; verification configuration, systemctl status nova-compute; systemctl status neutron-openvswitch-agent. The HA implementation effect of the proxy node is shown in Figure 6.


Status

Cluster Status: HEALTH_OK
Hosts: 3 total
Monitors: 3 (quorum 0, 1, 2)
OSDs: 6 total, 6 up, 6 in
Managers: 1 active, 1 standby
Object Gateways: 2 total
Metadata Servers: no filesystems
iSCSI Gateways: 0 total, 0 up, 0 down

Capacity

Raw Capacity: 0.07% of 300 GiB — Used: 206.2 MiB; Avail.: 299.8 GiB
Objects: 227 objects — Healthy: 100%; Misplaced: 0%; Degraded: 0%; Unfound: 0%
PG Status: 129 PGs — Clean: 129; Working: 0; Warning: 0; Unknown: 0
Pools: 5
PGs per OSD: 64.5

Performance

Client Read/Write: 0 IOPS — Reads: 0 /s; Writes: 0 /s
Client Throughput: 0 B/s — Reads: 0 B/s; Writes: 0 B/s
Recovery Throughput: 0 B/s
Scrubbing: Inactive

Figure 6. The HA implementation for proxy nodes.

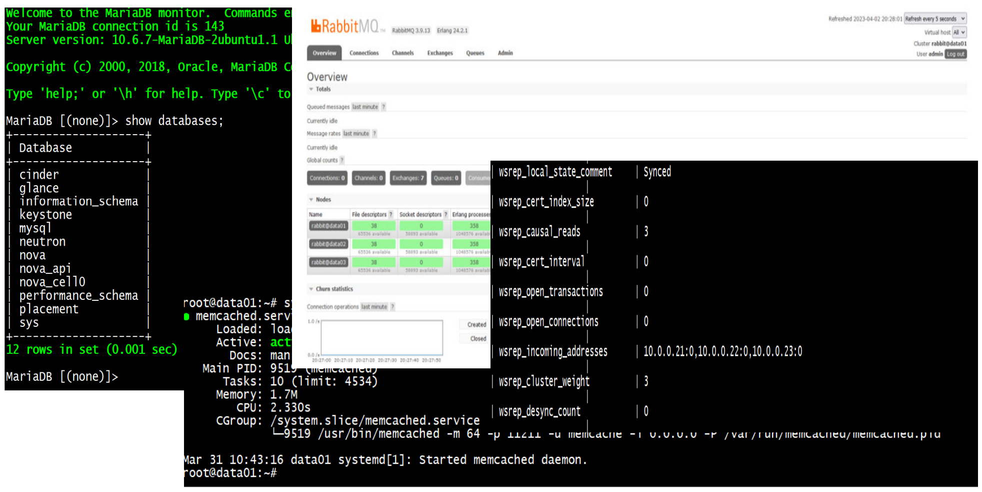
5.3. HA Implementation for Control Nodes

The control node servers are primarily comprised of two physical servers deploying core components such as Keystone, Glance, Placement, Nova, Neutron, Horizon, and Cinder. Keystone's deployment starts with component installation, followed by the creation of users, projects, and roles, and finally, the completion of node control deployment through identity authentication. When accessing the private cloud platform, identity authentication is performed first, and upon successful authentication, access to various OpenStack components is granted through tokens issued by Keystone. The management and renewal of these tokens are handled by HAProxy to ensure their validity and timeliness.

Glance is deployed either from the source code or through software packages, with the specification of the listening port and host IP for the Glance service. Similarly, HAProxy is installed from the source or packages, incorporating load balancing algorithms to distribute requests across different Glance servers, achieving HAProxy's load balancing service through the listening ports and host IPs specified by the backend.

In OpenStack, the HAProxy program provides load balancing for OpenStack Service APIs and MariaDB Galera services, with its own high availability implemented in an Active–Passive mode by Pacemaker+Corosync. Pacemaker is responsible for providing a Virtual IP (VIP) to HAProxy, ensuring that at any given time, only one HAProxy instance is in the Active state and providing services. The specific implementation steps are as follows:

Pcs resource create rip ocf: heartbeat: IPaddr2 params ip = "Server-VIP" cidr_netmask = "48" op monitor imerval = "30s", Pcs resource create lb-haproxy systemd: haproxy-clone, Pcs constraint order start vip then lb-haproxy-clone kind = Optional, Pcs constraint colocation add vip with lb-haproxyclone, the Memcached component inherently supports an Active–Active configuration, requiring only the configuration of all its node names within OpenStack, for example, memcached_servers = controller01:11211, controller02:11211. In the event that controller01: 11211 fails, the OpenStack service components will automatically switch to using controller02:11211. The HA implementation effect of the control node is shown in Figure 7.



```
Welcome to the MariaDB monitor.  Commands e
Your MariaDB connection id is 143
Server version: 10.6.7-MariaDB-2ubuntu1.1 U

Copyright (c) 2000, 2018, Oracle, MariaDB C

Type 'help;' or '\h' for help. Type '\c' to

MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| cinder             |
| glance             |
| information_schema |
| keystone           |
| mysql              |
| neutron            |
| nova               |
| nova_api           |
| nova_cell0         |
| performance_schema |
| placement          |
| sys                |
+--------------------+
12 rows in set (0.001 sec)

MariaDB [(none)]>
```

```
| wsrep_local_state_comment   | Synced
| wsrep_cert_index_size       | 0
| wsrep_causal_reads          | 3
| wsrep_cert_interval         | 0
| wsrep_open_transactions     | 0
| wsrep_open_connections      | 0
| wsrep_incoming_addresses    | 10.0.0.21:0,10.0.0.22:0,10.0.0.23:0
| wsrep_cluster_weight        | 3
| wsrep_desync_count          | 0
```

```
root@data01:~# s
● memcached.serv
     Loaded: loa
     Active: act
       Docs: man
   Main PID: 9519
      Tasks: 10 (limit: 4534)
     Memory: 1.7M
        CPU: 2.330s
     CGroup: /system.slice/memcached.service
             └─9519 /usr/bin/memcached -m 64

Mar 31 10:43:16 data01 systemd[1]: Started memcached daemon.
root@data01:~#
```

Figure 7. The HA implementation for control nodes.

5.4. HA Implementation for Data Nodes

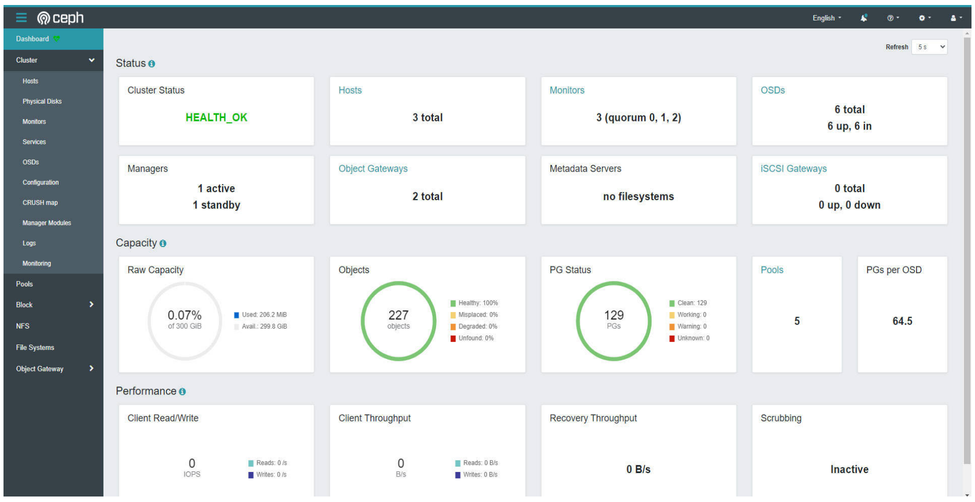
The data node servers deploy data cluster components, including the MariaDB high-availability database cluster, Memcached caching cluster, and RabbitMQ messaging queue cluster.

Multiple MariaDB instances are deployed across multiple servers, with each instance being a collection of databases and tables that share the same data and schema. Database replication is configured to utilize MariaDB's replication mechanism for master–slave replication, replicating updates from the primary database to standby databases. Additionally, a load balancer distributes client requests to achieve load balancing. Monitoring and failover mechanisms are implemented to monitor the health of each MariaDB instance and the performance of the load balancer. In case of an instance failure, the failover mechanism removes the failed instance from the cluster, initiates a standby instance, and makes it active.

A Memcached cluster is deployed across three physical servers with corresponding parameters set, such as memory size and port numbers. During cluster configuration, a consistent hashing algorithm is adopted to form a cluster of multiple Memcached nodes, where each node is responsible for caching a portion of the data. Node-to-node communication protocols, such as UDP or TCP, are also configured. The cluster's performance and stability are tested and monitored to ensure it can handle high volumes of concurrent requests and promptly identify and resolve issues.

A RabbitMQ cluster is deployed across three physical servers to work in coordination to ensure reliable message delivery. In a RabbitMQ cluster, each node operates independently, capable of receiving, storing, and forwarding messages. Nodes communicate over the network, utilizing Erlang's distributed system architecture for data synchronization and failure recovery. When setting up the cluster, it is crucial to ensure that the same version of RabbitMQ is installed on each node, and the node hostnames and IP addresses are configured correctly. Nodes hosting the service are joined to the cluster. Mirrored queues are configured to ensure message delivery even in case of node failures, and load balancing is set up to distribute messages evenly among nodes. After all deployments and configurations are complete, queue messaging is used to test the accuracy of information transmission within the RabbitMQ cluster. The implementation steps are as follows:

Install Cinder and Ceph on the data node, apt install cinder-volume ceph ceph-common; configure Cinder to use Ceph as the backend, /etc/cinder/cinder.conf, volume_driver = cinder.volume.drivers.rbd.RBDVolumeDriver, volume_backend_name = ceph, rbd_pool = ZLC_volumes, rbd_user = ZLC_cinder, rbd_secret_uuid = <secret_uuid>, ceph_conf = /etc/ceph/ceph.conf, systemctl restart cinder-volume, journalctl -u cinder-volume. The HA implementation effect of the data node is shown in Figure 8.



Status

Cluster Status: HEALTH_OK
Hosts: 3 total
Monitors: 3 (quorum 0, 1, 2)
OSDs: 6 total, 6 up, 6 in
Managers: 1 active, 1 standby
Object Gateways: 2 total
Metadata Servers: no filesystems
iSCSI Gateways: 0 total, 0 up, 0 down

Capacity

Raw Capacity: 0.07% of 300 GiB; Used: 206.2 MiB; Avail.: 299.8 GiB
Objects: 227 objects; Healthy: 100%; Misplaced: 0%; Degraded: 0%; Unfound: 0%
PG Status: 129 PGs; Clean: 129; Working: 0; Warning: 0; Unknown: 0
Pools: 5
PGs per OSD: 64.5

Performance

Client Read/Write: 0 IOPS; Reads: 0 /s; Writes: 0 /s
Client Throughput: 0 B/s; Reads: 0 B/s; Writes: 0 B/s
Recovery Throughput: 0 B/s
Scrubbing: Inactive

Figure 8. The HA implementation for data node.

5.5. HA Implementation for Compute Nodes

To meet diverse teaching requirements, the private cloud platform utilizes three physical machines as compute nodes to provide highly available, elastic, and scalable computing resource services.

In terms of networking, the platform has chosen the FLAT and VXLAN networks. The FLAT network is a simple Layer 2 network that enables communication between virtual machines and physical machines. The VXLAN network, on the other hand, is a virtualization network based on tunneling technology, suitable for expansion across data centers. For storage, distributed storage Ceph is employed to provide functions such as object storage, block storage, and file system storage.

In addition to networking and storage technologies, optimizations have been carried out on the compute nodes to enhance their stability and reliability. These optimizations primarily focus on optimizing virtual machine configuration and management, physical machine storage and monitoring, as well as fault tolerance and backup mechanisms for the entire compute node cluster. Through these optimization efforts, efficient and stable computing resource services can be provided, fulfilling the flexible and efficient usage needs of high-quality teaching resources for both students' learning and teachers' instruction.

To achieve live migration of virtual machine instances using Ceph distributed storage, we leverage Ceph as the distributed file system to ensure high availability across compute nodes. With Ceph Monitor installed and configured on the Controller node, and Ceph-Client installed and configured on the Compute nodes, we can build an HA implementation effect of the compute node, as shown in Figure 9. The specific configuration statements would be outlined as follows:

Live_migration_flag = vir_migrate_undefine_source, vir_migrate_peer2peer, vir_migrate_live, vir_migrate_tunnelled.

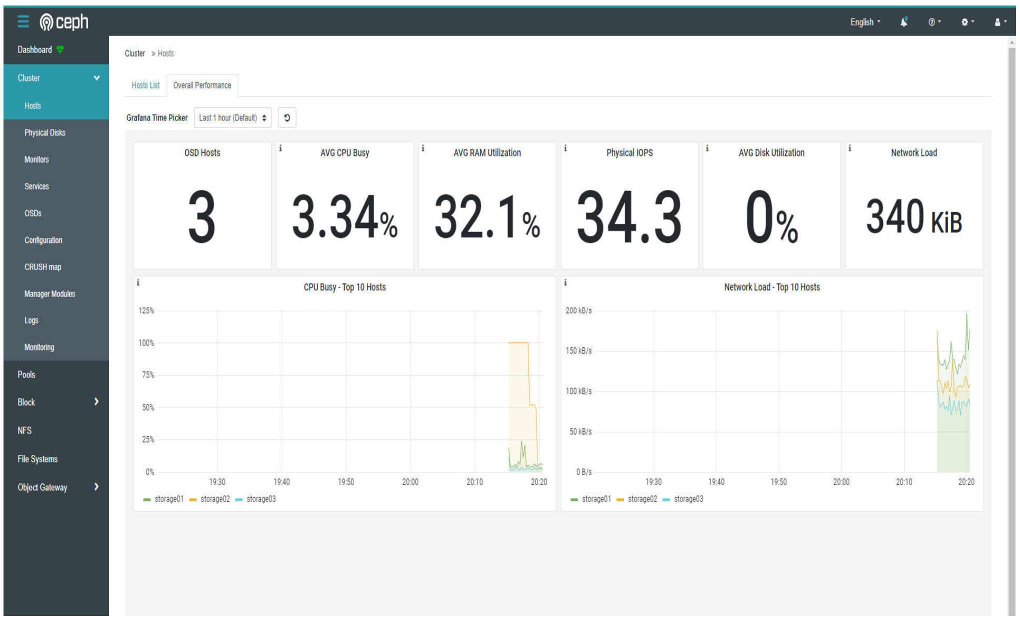


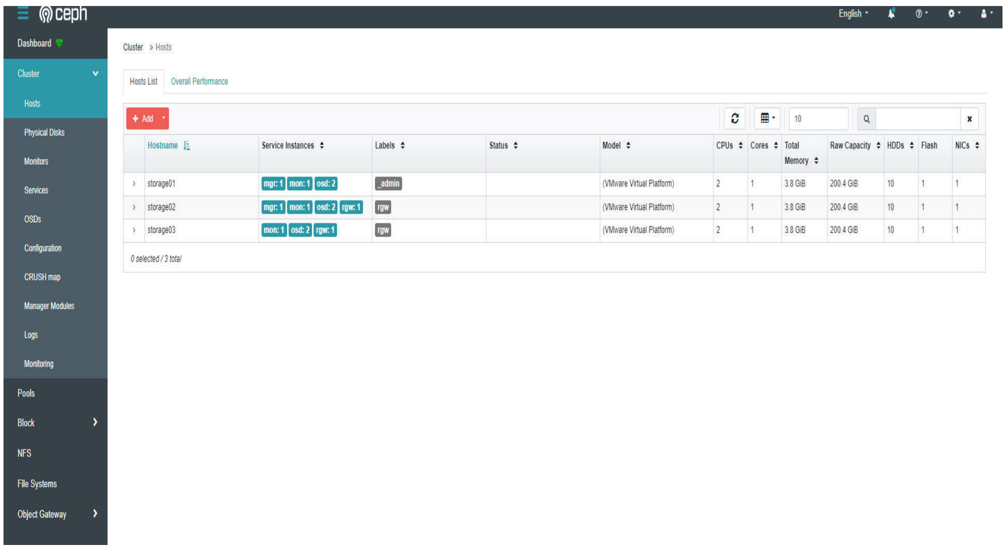
Figure 9. The HA implementation for compute nodes.

5.6. HA Implementation for Storage Nodes

The distributed storage system combining OpenStack with Ceph realizes the integration and unified management of storage resources across multiple nodes. OpenStack + Ceph provides the private cloud platform with object storage, block storage, and file system storage, achieving the efficient integration of various storage resources.

The deployment of storage resources begins with the installation of the Ceph RADOS Block Device driver, followed by the construction of the Ceph storage cluster. The Ceph storage cluster comprises multiple distributed storage nodes, with each storage node running in the ceph-osd process. The Ceph-core daemon ensures high availability and scalable storage resource services. During the deployment of Ceph storage nodes, storage pools are also created to store data such as virtual machine images, volumes, and objects. These storage pools not only allow specifying different storage policies but also enable features like replicas, snapshot replicas, and data protection, providing users with elastic and scalable storage services. The implementation steps are as follows:

Configure Cinder to use Ceph as a high-performance block storage backend, /etc/cinder/cinder.conf, enabled_backends = ceph, volume_driver = cinder.volume.drivers.rbd.RBDVolumeDriver, volume_backend_name = ceph, rbd_pool = volumes, rbd_user = cinder, rbd_secret_uuid = <secret_uuid> ceph auth get-key client.cinder, ceph_conf = /etc/ceph/ceph.conf, rbd_flatten_volume_from_snapshot = false, rbd_max_clone_depth = 5, rbd_store_chunk_size = 4. The HA implementation effect of the storage node is shown in Figure 10.



	Hostname	Service Instances	Labels	Status	Model	CPUs	Cores	Total Memory	Raw Capacity	HDDs	Flash	NICs
›	storage01	mgr: 1 mon: 1 osd: 2	_admin		(VMware Virtual Platform)	2	1	3.8 GiB	200.4 GiB	10	1	1
›	storage02	mgr: 1 mon: 1 osd: 2 rgw: 1	rgw		(VMware Virtual Platform)	2	1	3.8 GiB	200.4 GiB	10	1	1
›	storage03	mon: 1 osd: 2 rgw: 1	rgw		(VMware Virtual Platform)	2	1	3.8 GiB	200.4 GiB	10	1	1

0 selected / 3 total

Figure 10. The HA implementation for storage nodes.

5.7. HA Implementation for Network Nodes

Neutron's high availability utilizes the native high-availability modes provided by OpenStack, implementing different high-availability approaches for various services. The high-availability architecture for network nodes is achieved by Neutron components. These methods rely on Neutron components to implement L2 and L3 layer agents and plugins for networking.

The L2 agent only provides services on the network or compute node where it resides; therefore, it does not require HA. The DHCP Agent, being a scheduler of OpenStack networking services, allows running agents across multiple nodes. The DHCP protocol inherently supports multiple DHCP servers. Thus, multiple DHCP Agents can be created for each tenant network by configuring the number of DHCPs per network on multiple network controller nodes, by modifying the dhcp_agents_per_network parameter in the /etc/neutron/neutron.conf file to 2 or more.

For LBaaS (Load Balancer as a Service) high availability, it cannot be achieved through the HAProxy plugin that comes with Neutron LBaaS. Instead, HAProxy's high-availability solution leverages the Virtual Router Redundancy Protocol (VRRP). Therefore, an Active–Passive LBaaS Agent HA deployment is implemented using Pacemaker + DRBD (for the /var/lib/neutron/lbaas/ directory).

There are three ways to achieve Neutron L3 high availability: (1) utilize Pacemaker+ corosync to implement Active–Passive high availability; (2) for compatibility with high availability and scalability, employ VRRP-based Active–Active high availability; (3) implement Neutron L3 high availability using Distributed Virtual Routing (DVR) introduced in the Juno release. With DVR, this solution deploys DHCP and SNAT only on the network control nodes, while deploying NAT and L3 Agent to the compute nodes where VMs reside. This simultaneously resolves the high-availability issues of L3 Agent and Metadata Agent, with specific configurations outlined in Table 2 and the HA implementation effect of the network node is shown in Figure 11.

Table 2. Distributed Virtual Routing (DVR) Configuration.

Nodes	Configuration
Proxy Nodes	/etc/nova/nova.conf/etc/neutron/plugins/ml2/ml2_conf.ini
Control Nodes	/etc/neutron/neutron.conf router_distributed = True
Data Nodes	/etc/cinder/cinder.conf/etc/ceph/ceph.conf systemctl restart cinder-volume
Compute Nodes	/etc/neutron/l3_agent.ini/etc/neutron/plugins/ml2/ml2_conf.ini agent_mode = dvr enable_distributed_routing = True

Table 2. *Cont.*

Nodes	Configuration
Storage Nodes	/etc/cinder/cinder.conf rbd_flatten_volume_from_snapshot = false
Network Nodes	/etc/neutron/l3_agent.ini/etc/neutron/plugins/ml2/ml2_conf.ini agent_mode = dvr_snat enable_distributed_routing = True

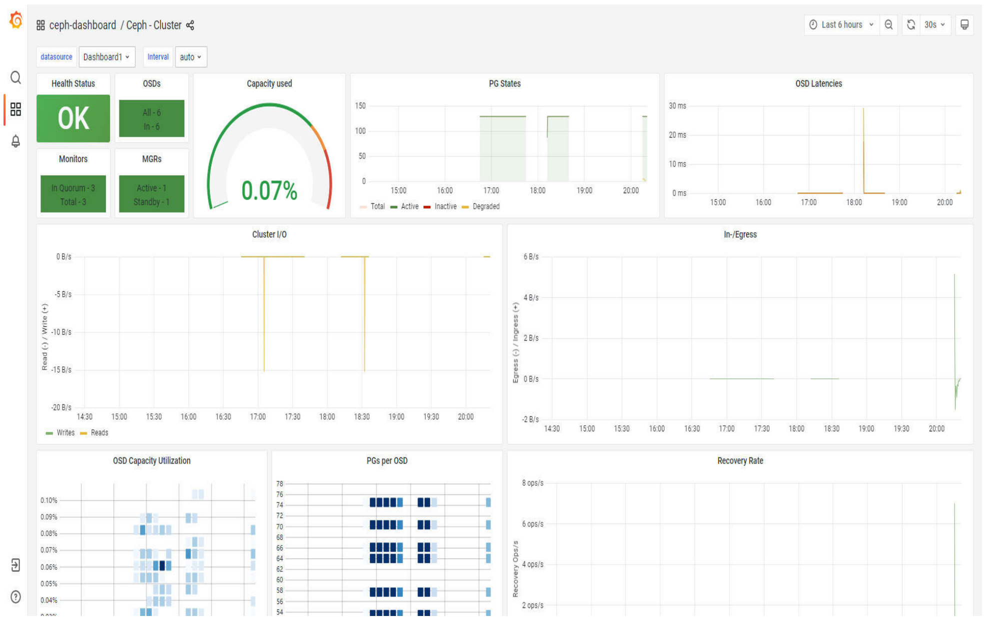


Figure 11. The HA implementation for network nodes.

6. Test and Analysis of the Private Cloud Platform for Educational Resources

The private cloud education platform built with OpenStack + Ceph can provide customized educational resource services, including tailored Learning Management Systems (LMS), course content, teaching tools, and assessment methods. Furthermore, it allows educational institutions or users to control the storage and processing of their data, ensuring compliance with data protection regulations. Disconnecting any network cable from a relevant node of the OpenStack cloud platform triggers warnings while the platform continues to function normally. Upon restoring the network, the OpenStack cloud platform's warnings disappear, and the platform resumes normal operations. Disconnecting any load balancer node ensures that the OpenStack cloud platform can still provide services normally, indicating the absence of a single point of failure (SPoF). Disconnecting any OpenStack cloud platform control node reveals that the service components, databases, and messaging queue clusters associated with the control node can all continue to provide normal services. Disconnecting a small number of nodes from the Ceph cluster maintains the smooth operation and stability of the virtual machine instances within the OpenStack cloud platform.

Compared to a private cloud education platform built with OpenStack + Swift, the OpenStack + Ceph platform utilizes simpler modules, handles data backups more effectively, offers better OSD location addressing, supports unified storage, boasts high reliability, and exhibits a high degree of automation and scalability. The replicas in the OpenStack + Swift platform are written asynchronously, which can potentially lead to incomplete updates and the reading of incorrect data. Moreover, the architectural design of OpenStack + Swift may not be as strong in terms of transfer speed and latency. The proxy server nodes in Swift also have traffic limits, and apart from object storage functionality, it does not provide block storage or file storage capabilities.

Compared to cloud platforms such as Google Cloud Platform (GCP) and Amazon Web Services (AWS), the cloud platform built on OpenStack and Ceph in this context offers superior performance, reliability, flexibility, and scalability, along with multi-user support and resource isolation, rich service and integration capabilities, community support, and continuous improvement. However, both OpenStack and Ceph are complex systems comprising multiple components and modules, requiring regular updates and maintenance to ensure stability and security. It is crucial to closely monitor security issues and implement corresponding security measures to mitigate risks.

In summary, the educational resource private cloud platform built by combining OpenStack + Ceph boasts significant advantages such as high performance, reliability, flexibility, and scalability. However, it also presents potential drawbacks, including complexity, maintenance costs, challenges in performance and stability, and security risks. When implementing this platform, it is essential to comprehensively consider and evaluate one's own business requirements and actual situation.

7. Conclusions

The private cloud platform built with OpenStack + Ceph features high availability, capable of detecting and handling single points of failure and service instance failures, automatically switching between service nodes, and achieving high-availability services for the OpenStack cloud platform. It can rapidly switch service instance resources between high-availability nodes, ensuring uninterrupted cloud platform services and stable virtual machine instances. Compared to a private cloud platform built with OpenStack + Swift, OpenStack + Ceph can reduce deployment time by 50%, enhancing computing speed. Through Ceph technology, it achieves up to three 9s (99.9%) data reliability, which translates to only 52.6 min of downtime per year, significantly improving data reliability and integrity.

OpenStack + Ceph also supports rapid horizontal scaling, enabling the addition of several compute nodes within hours to meet the resource demands of students and teachers. The automated management of OpenStack + Ceph can enhance resource utilization, reducing resource waste by up to 20%. Furthermore, through optimization of OpenStack and Ceph, the response time for user requests can be significantly reduced, achieving a decrease from 10 s to 2 s compared to optimizations with OpenStack and Swift. The combination of OpenStack and Ceph can also improve service availability, minimizing service interruption time to less than 5 min per year. As a result, the pairing of OpenStack with Ceph provides educational institutions with a high-performance, highly reliable, and elastically scalable private cloud service, catering to the flexible and efficient utilization requirements of high-quality teaching resources needed for student learning and teacher guidance.

As educational resources continue to proliferate and become more complex, the future holds the potential to further enhance storage performance by incorporating additional high-performance underlying technologies such as SPDK and BlueStore, thereby satisfying the efficient read–write requirements of large-scale educational data. Concurrently, the resource scheduling and management capabilities of OpenStack can be optimized to allocate computing, networking, and storage resources more intelligently, ultimately improving the overall platform's resource utilization and response speed.

Author Contributions: Conceptualization, L.Z.; data curation, Y.X.; formal analysis, G.H.; investigation, L.Z.; methodology, L.Z.; resources, G.H.; software, G.H.; validation, Y.X.; writing—original draft, Y.X. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the Guiyang City Science and Technology Plan Project (NO.[2024]2-22), the Scientific Studies of Higher Education Institution, Guizhou Province Education Department(Youth Project, QEJ[2022]307, QEJ[2021]005), the Science and Technology Foundation of Guizhou Province (QKHJC-ZK[2023]012, QKHJC-ZK[2022]014), the Doctoral Research Start-up Fund of Guiyang University (GYU-KY-2022), the Shenzhen Natural Science Fund (JCYJ20200109144601715, JCYJ20200109144612399), and IER Foundation 2023 (IERF202304).

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to privacy or ethical restrictions.

Acknowledgments: The authors would like to thank the anonymous reviewers for their comments that helped improve this paper.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Tlili, A.; Zhang, J.; Papamitsiou, Z.; Manske, S.; Huang, R.; Kinshuk; Hoppe, H.U. Towards utilising emerging technologies to address the challenges of using Open Educational Resources: a vision of the future. *Educ. Tech Res. Dev.* **2021**, *69*, 515–532. [CrossRef]
2. Prameela, P.K.; Gadagi, P.; Gudi, R.; Patil, S.; Narayan, D.G. Energy-efficient VM management in OpenStack-based private cloud. In *Advances in Computing and Network Communications: Proceedings of CoCoNet 2020*; Springer: Singapore, 2021; Volume 1, pp. 541–556. [CrossRef]
3. Sai, Y.; Zhang, T. Analysis of key technologies of cloud computing based on openstack cloud platform. In Proceedings of the International Conference on Mathematics, Modeling, and Computer Science (MMCS2023), SPIE, Belgrade, Serbia, 28–31 August 2023; Volume 12625, pp. 567–572. [CrossRef]
4. Abbasi, M.; Cardoso, F.; Silva, J.; Martins, P. Exploring OpenStack for Scalable and Cost-Effective Virtualization in Education. In *International Conference on Disruptive Technologies, Tech Ethics and Artificial Intelligence*; Springer: Cham, Switzerland, 2023; p. 135. [CrossRef]
5. Maaz, M.; Ahmed, M.A.; Maqsood, M.; Soma, S. Development Of Service Deployment Models In Private Cloud. *J. Sci. Res. Technol.* **2023**, *1*, 1–12. [CrossRef]
6. Bhatia, G.; Al Noutaki, I.; Al Ruzeiqi, S.; Al Maskari, J. Design and implementation of private cloud for higher education using OpenStack. In Proceedings of the 2018 Majan International Conference (Mic), Muscat, Oman, 19–20 March 2018; pp. 1–6. [CrossRef]
7. Khan, H.M.; Cerveira, F.; Cruz, T.; Madeira, H. Network Failures in Cloud Management Platforms: A Study on OpenStack. In Proceedings of the 13th International Conference on Cloud Computing and Services Science, Prague, Czech Republic, 26–28 April 2023; pp. 228–235. [CrossRef]
8. Lame, N.; Adding a Dynamic Load Balancing Based on a Static Method in Cloud via OpenStack. École de Technologie Supérieure. 2023. Available online: https://espace.etsmtl.ca/id/eprint/3293 (accessed on 3 July 2024).
9. SM, H.K.; Sharma, R. Improving Orchestration Service Using gRPC API and P4-Enabled SDN Switch in Cloud Computing Platform: An OpenStack Case. *IAENG Int. J. Comput. Sci.* **2023**, *50*, 1–15. Available online: https://api.semanticscholar.org/CorpusID:264552335 (accessed on 3 July 2024).
10. Benomar, Z.; Longo, F.; Merlino, G.; Puliafito, A. Cloud-based network virtualization in iot with openstack. *ACM Trans. Internet Technol. (TOIT)* **2021**, *22*, 1–26. [CrossRef]
11. Aregbesola, M.K.; Aro, T.O.; Aiyeniko, O.; Olukiran, O.O.; Akanni, O.O. Open-Source Cloud Computing Platforms: Comparative Study. *Adeleke Univ. J. Eng. Technol.* **2022**, *5*, 125–136. [CrossRef]
12. Mane, A.S.; Ainapure, B.S. Private Cloud Configuration Using Amazon Web Services. In *Information and Communication Technology for Competitive Strategies (ICTCS 2021) Intelligent Strategies for ICT*; Springer Nature: Singapore, 2021; pp. 839–847. [CrossRef]
13. Faraji Shoyari, M.; Ataie, E.; Entezari-Maleki, R.; Movaghar, A. Availability modeling in redundant OpenStack private clouds. *Softw. Pract. Exp.* **2021**, *51*, 1218–1241. [CrossRef]
14. Kai, Z.; Youyu, L.; Qi, L.; Hao, S.C.; Liping, Z. Building a private cloud platform based on open source software OpenStack. In Proceedings of the 2020 International Conference on Big Data and Social Sciences (ICBDSS), IEEE, Xi'an, China, 14–16 August 2020; pp. 84–87. [CrossRef]
15. Lima, S.; Rocha, A.; Roque, L. An overview of OpenStack architecture: a message queuing services node. *Clust. Comput.* **2019**, *22*, 7087–7098. [CrossRef]
16. Gaikwad, C.; Churi, B.; Patil, K.; Tatwadarshi, P.N. Providing storage as a service on cloud using OpenStack. In Proceedings of the 2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), IEEE, Coimbatore, India, 17–18 March 2017; pp. 1–4. [CrossRef]
17. Zhang, J.; Zhang, J.; Ding, H.; Wan, J.; Ren, Y.; Wang, J. Designing and Applying an Education IaaS System based on OpenStack. *Appl. Math. Inf. Sci.* **2013**, *7*, 155–160. Available online: https://api.semanticscholar.org/CorpusID:15714624 (accessed on 3 July 2024). [CrossRef]
18. Lima, S.; Rocha, A. A View of OpenStack: Toward an Open-Source Solution for Cloud. *World Conf. Inf. Syst. Technol.* **2017**, *3*, 481–491. [CrossRef]
19. Wagh, N.; Pawar, V.; Kharat, K. Implementation of Stable Private Cloud using OpenStack with Virtual Machine Results. *J. Comput. Eng. Technol.* **2019**, *10*, 258–269. [CrossRef]
20. Bonner, S.; Pulley, C.; Kureshi, I.; Holmes, V.; Brennan, J.; James, Y. Using OpenStack to improve student experience in an H.E. environment. In Proceedings of the Science and Information Conference, IEEE, London, UK, 7–9 October 2013; pp. 888–893. Available online: https://api.semanticscholar.org/CorpusID:13303143 (accessed on 3 July 2024).
21. Virupakshar, K.B.; Asundi, M.; Channal, K.; Shettar, P.; Patil, S.; Narayan, D.G. Distributed denial of service (DDoS) attacks detection system for OpenStack-based private cloud. *Procedia Comput. Sci.* **2020**, *167*, 2297–2307. [CrossRef]

22. Kominos, C.G.; Seyvet, N.; Vandikas, K. Bare-metal, virtual machines and containers in OpenStack. In Proceedings of the 2017 20th Conference on Innovations in Clouds, Internet and Networks (ICIN), IEEE, Paris, France, 7–9 March 2017; Volume 1, pp. 36–43. [CrossRef]
23. Bystrov, O.; Pacevič, R.; Kačeniauskas, A. Performance of communication-and computation intensi-ve SaaS on the OpenStack cloud. *Appl. Sci.* **2021**, *11*, 7379. Available online: https://etalpykla.vilniustech.lt/handle/123456789/111573 (accessed on 4 July 2024). [CrossRef]
24. Han, Z.; Heng, Y.; Fang, W. Research on the Application of OpenStack+Ceph Cloud Storage Technology. *J. Huaibei Vocat. Tech. Coll.* **2024**, *23*, 113–116. Available online: https://link.cnki.net/10.16279/j.cnki.cn34-1214/z.2024.03.021 (accessed on 4 July 2024).
25. Qian, Y.; Lehman, J. Students' Misconceptions and Other Difficulties in Introductory Programming: A Literature Review. *ACM Trans. Comput. Educ.* **2017**, *18*, 1–24. [CrossRef]
26. Xi, L. Design and Implementation of Ceph Distributed Storage System in OpenStack Cloud Platform. *Digit. Media Res.* **2021**, *38*, 37–43. Available online: https://link.cnki.net/10.16279/j.cnki.cn34-1214/z.2021.08.081 (accessed on 4 July 2024).
27. Sun, S. A Solution for Building a Private Cloud Based on OpenStack and Ceph. *Inf. Rec. Mater.* **2021**, *22*, 157–160. Available online: https://link.cnki.net/10.16009/j.cnki.cn13-1295/tq.2021.01.108 (accessed on 4 July 2024).
28. Zhang, Y. Design and Implementation of an Open Source Private Cloud Platform Based on OpenStack and Ceph. *Yunnan Electr. Power Technol.* **2019**, *47*, 61–63+68. Available online: https://link.cnki.net/10.16009/j.cnki.cn13-1395/tq.2019.02.047 (accessed on 4 July 2024).
29. Toor, S.; Osmani, L.; Eerola, P.; Kraemer, O.; Lindén, T.; Tarkoma, S.; White, J. A scalable infrastructure for CMS data analysis based on OpenStack Cloud and Gluster file system. *J. Phys. Conf. Ser.* **2014**, *513*, 062047. [CrossRef]
30. Van, V.N.; Chi, L.M.; Long, N.Q.; Nguyen, G.N.; Le, D.N. A performance analysis of openstack open-source solution for IaaS cloud computing. In Proceedings of the Second International Conference on Computer and Communication Technologies: IC3T, 2015; Springer: New Delhi, India, 2016; Volume 2, pp. 141–150. [CrossRef]
31. Lu, M.; Zhou, X. A big data on private cloud agile provisioning framework based on OpenStack. In Proceedings of the 2018 IEEE 3rd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA), IEEE, Chengdu, China, 20–22 April 2018; pp. 253–260. [CrossRef]
32. Kabiri, M.N.; Wannous, M. An experimental evaluation of a cloud-based virtual computer laboratory using openstack. In Proceedings of the 2017 6th IIAI International Congress on Advanced Applied Informatics (IIAI-AAI), IEEE, Hamamatsu, Japan, 9–13 July 2017; pp. 667–672. [CrossRef]
33. Ismail, M.A.; Ismail, M.F.; Ahmed, H. Openstack cloud performance optimization using linux services. In Proceedings of the 2015 International Conference on Cloud Computing (ICCC), IEEE, Riyadh, Saudi Arabia, 26–29 April 2015; pp. 1–4. [CrossRef]
34. Kang, M.; Kang, D.I.; Walters, J.P.; Crago, S.P. A comparison of system performance on a private openstack cloud and amazon ec2. In Proceedings of the 2017 IEEE 10th International Conference on Cloud Computing (CLOUD), IEEE, Honololu, HI, USA, 25–30 June 2017; pp. 310–317. [CrossRef]
35. Yunxia, J.; Bowen, Z.; Shuqi, W.; Dongnan, S. Research of enterprise private cloud computing platform based on openstack. *Int. J. Grid Distrib. Comput.* **2014**, *7*, 171–180. [CrossRef]
36. Raja, J.B.; Rabinson, K.V. IAAS for private and public cloud using Openstack. *Int. J. Eng.* **2016**, *5*, 5–7. [CrossRef]
37. Mangal, G.; Kasliwal, P.; Deshpande, U.; Kurhekar, M.; Chafle, G. Flexible cloud computing by integrating public-private clouds using openstack. In Proceedings of the 2015 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), IEEE, Bangalore, India, 25–27 November 2015; pp. 146–152. [CrossRef]
38. Kumar, R.; Gupta, N.; Charu, S.; Jain, K.; Jangir, S.K. Open source solution for cloud computing platform using OpenStack. *Int. J. Comput. Sci. Mob. Comput.* **2014**, *3*, 89–98. [CrossRef]
39. Sefraoui, O.; Aissaoui, M.; Eleuldj, M. OpenStack: Toward an Open-Source Solution for Cloud Computing. *Int. J. Comput. Appl.* **2012**, *55*, 38–42. [CrossRef]
40. Grzonka, D. The analysis of openstack cloud computing platform: Features and performance. *J. Telecommun. Inf. Technol.* **2015**, *3*, 52–57. [CrossRef]
41. Tissir, N.; ElKafhali, S.; Aboutabit, N. How much your cloud management platform is secure? OpenStack Use Case. In *Innovations in Smart Cities Applications Volume 4: The Proceedings of the 5th International Conference on Smart City Applications, Karabuk, Turkey, 7–9 October 2020*; Springer International Publishing, Springer: Cham, Switzerland, 2021; pp. 1117–1129. [CrossRef]