



A High-Availability Cloud for Research Computing

Justin Riley, John Noss, Wes Dillingham, and James Cuff,
Harvard University

Ignacio M. Llorente, OpenNebula, Harvard University,
and Complutense University

This article describes the lessons learned, challenges faced, and innovations made in designing and implementing a high-availability private cloud for research computing.

FAS RC, the Research Computing group in Harvard University's Faculty of Arts and Sciences Division of Science, facilitates the advancement of complex research by providing leading-edge services for high-performance and scientific computing, bioinformatics analysis, visualization, and data storage.

Recently, we designed a strategy to convert FAS RC's legacy internal KVM (kernel-based virtual machine) infrastructure from a homemade virtualized cluster dependent on scripted tools to a more robust, reliable, and automated private-cloud system. We then configured the system to enable integration with the public cloud to improve agility, increase resource utilization, and allow implementation of further advanced services.

Our first step was designing a cloud reference architecture with high availability, multitenancy, orchestration, and provisioning to yield a common frame of reference for all private-cloud instances. This provided a foundation for further development and innovation, and helped us make well-founded strategic and technical decisions. The architecture design provides APIs and features that help serve users more efficiently. These features also better enable us to test new configurations and dynamically increase resources for continuous integration and deployment.

To ensure the strategy's success, we had to plan and automate the clouds' operations and maintenance processes and then integrate them with existing datacenter- and configuration-management tools. The cloud-infrastructure deployment is fully automated and provides three fault-tolerant infrastructures with a multitiered backup



system and robust VM and virtual disk monitoring:

- › a core cloud infrastructure to simplify internal management and enhance hosted services' resilience;
- › a science cloud to allow Harvard's scientific community to lease resources; and
- › a testing cloud to enable testing and staging changes, as well as experimentation with infrastructure-as-a-service computing.

This article describes the lessons learned, challenges faced, and innovations made in the design and implementation of this system to guide other organizations transitioning to a private cloud and automated infrastructure.

ARCHITECTURE DESIGN FOR HIGH AVAILABILITY

High availability (HA) is a cloud's ability to keep functioning after one or more hardware or software components fail.¹ We achieved HA by incorporating features such as redundancy for failover and replication for load balancing in each component without using costly specialized hardware and software.

Our reference architecture, shown in Figure 1, employs a classic cluster-like organization² with a controller host, a set of hypervisors for hosting VMs, a storage cluster, and at least one physical network connecting the hosts. The architecture uses OpenNebula³ as the orchestration manager, Ceph⁴ as the storage cluster, KVM as the hypervisor, and Microsoft Azure and Amazon Web Services as two public clouds for *cloud-bursting*. The architecture is functional across two datacenters. This lets VMs migrate between them for load balancing or if one datacenter needs maintenance or experiences problems.

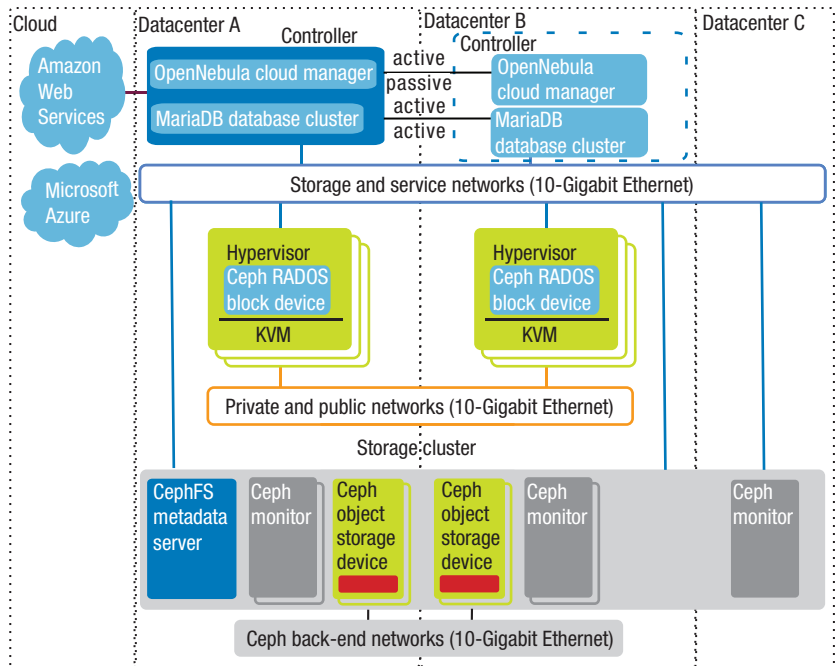


Figure 1. Internal private cloud architecture based on three datacenters with hybrid access to public clouds. High-availability controller hosts run in the two primary datacenters (A and B). Ceph's object-storage devices span the primary datacenters to remain active in case of datacenter failure. Datacenter C hosts a fifth Ceph monitor and facilitates negotiating cluster consensus during and after potential datacenter failures. KVM: kernel-based virtual machine.

Two controller hosts run an HA active-passive setup of the OpenNebula cloud manager and an HA active-active configuration of a MariaDB database cluster. The OpenNebula front ends have fencing mechanisms so that if one instance fails, the system takes the node offline and the passive node acquires its IP address to avoid service disruptions. The MariaDB cluster keeps all data related to cloud objects—such as hosts, networks, VMs, and users—synchronized between controller nodes so that all data is accessible if a node fails.

Hypervisor hosts provide VMs with execution resources, such as CPU, memory, and network access. Users can configure the KVM-based hypervisor nodes via OpenNebula with an automated VM restart, which enables

the system to prepare for and recover from VM or host failures. If one physical host fails, the system can redeploy all of its VMs on another host. If a VM crashes, the system restarts it.

A dedicated Ceph cluster provides cloud storage. We configured the CRUSH (controlled replication under scalable hashing) data-placement algorithm to distribute four replicas per object between two hosts in each of the two datacenters containing Ceph object storage devices (OSDs). This replication strategy ensures that the Ceph cluster can tolerate the loss of an entire datacenter without going into a read-only state and the loss of a single host in the surviving datacenter without losing data.

Two Ceph monitors, which use a Paxos protocol to establish a quorum

consensus about cluster state, are in each datacenter alongside the OSDs and a fifth monitor is in a third datacenter. This fifth monitor helps maintain the quorum in case a link between datacenters is lost or a datacenter fails. In addition to providing block devices, Ceph's CephFS POSIX file system provides storage for OpenNebula's files. A standby node in each datacenter

both custom software and OpenNebula hooks to meet internal requirements for activities such as backups, monitoring, and alerting. For example, we developed

- › onedns, a dynamic OpenNebula-aware DNS server;
- › a VM hook to prevent OpenNebula from causing existing

from Ceph's client-side admin socket to Graphite for each VM disk. This provides a leaderboard that shows real-time disk I/O from each running VM without relying on a client running in the VM. This lets FAS RC administrators quickly detect VMs that are hurting the storage cluster's performance.

We developed a backup process utilizing Ceph's per-block-device snapshot mechanism. Each day, the system compares the current and previous day's snapshots, and exports the differing objects to a separate backup cluster. They are then applied to another block device, thereby providing incremental snapshots on an isolated cluster. We also developed a script (ceph-rsnapshot) to convert Ceph RBDs to the qcow2 file format and ship them to our backup file system in a separate datacenter. We run this from the backup Ceph cluster to minimize the production cluster's load when converting RBDs to qcow2 backups. An additional script exports the OpenNebula templates as XML files hourly so that FAS RC administrators can reference them or use them for a recovery operation.

To automate the deployment and configuration of OpenNebula and Ceph, we use Puppet, the open source configuration-management tool already utilized at FAS RC. This enables the provisioning and management of multiple OpenNebula clouds with a single toolset and also provides a common place for configuring the system with the rest of the FAS RC infrastructure.

SCIENCE-CLOUD RESOURCE PROVISIONING

Our science-cloud infrastructure will serve multiple user groups. An on-premise private cloud in a large organization like Harvard requires powerful, flexible mechanisms to manage access privileges to the virtual and physical infrastructure and also to dynamically allocate available resources.

The planned resource-provisioning model defines a virtual datacenter for

As users move their workloads to cloud services to process large data volumes, we must rethink their architectures to focus on data storage and management.

makes the CephFS metadata server highly available.

Before placing the storage system into production, we thoroughly tested it under various failure scenarios (such as a failed disk or a network-connection loss) and under heavy loads. We also configured sufficient spare capacity on controllers, hypervisors, and storage hosts so that while they are being maintained, administrators can live-migrate VMs off of a hypervisor or take down Ceph OSDs and monitors without service downtime.

Each host has two Link Aggregation Control Protocol-bonded 10-Gigabit Ethernet network interface controllers connected to two switches for HA. The cloud uses several IEEE 802.1q-based virtual LANs for service, storage, and public and private communications. Using OpenNebula security groups, administrators can set firewall rules and apply them to VMs, thereby defining permitted inbound and outbound traffic.

CLLOUD DEPLOYMENT AND OPERATION

The reference architecture's deployment and operation entailed several challenges.

Integrating the new private-cloud environments into the existing FAS RC infrastructure required us to write

physical systems on shared VLANs to experience IP conflicts;

- › tools for monitoring the I/O load of individual Ceph VM disks;
- › tools such as ceph-rsnapshot to back up Ceph RADOS block device (RBD) images via the rsnapshot file-system snapshot utility to the multitiered backup system; and
- › procedures for supporting various types of nontrivial VMs, such as license servers that require specific MAC addresses to function properly.

The Ceph cluster is monitored using the Ceph-dash open source utility; a flask-based API; and a web dashboard, which monitors the cluster's overall status and issues Nagios alerts for problems that are encountered. Diamond collectors gather cluster and component metrics, and the Graphite plug-in for Grafana stores and graphs the metrics.

One of the legacy KVM infrastructure's big problems was not having access to per-VM disk I/O statistics, making it hard to identify VMs with high I/O loads. We extended the Ceph Diamond collector to query the OpenNebula front end for VMs on the hypervisors and then ship metrics

each user group, isolated from those of the other groups. The model also dynamically allocates resources based on need and assigns resource quotas. Each group will have its own internal administrator and has access to a private image catalog for its research field that can be easily consumed via a self-service portal. This model will evolve and improve as it incorporates additional features.


TOWARD A DATA-CENTRIC APPROACH TO CLOUD COMPUTING

Recently, the US National Science Foundation funded the North East Storage Exchange (projectnese.org) to create a multipetabyte object store that will be designed, built, and supported by the same university partners that founded the Massachusetts Green High-Performance Computing Center (www.mghpcc.org). This storage-cloud infrastructure will tightly integrate with the cloud environments at FAS RC, to provide research faculty with on-demand services for data-intensive applications supporting Harvard's new Data Science Initiative (datascience.harvard.edu).

As users move their workloads to cloud services to process large data volumes, we must rethink their architectures to focus on data storage and management. This paradigm shift in cloud design from computation to data exploration requires infrastructures to be data-centric. This would minimize data movement—which increases network overhead, data-access latency, and energy consumption—by bringing computation closer to data within the architecture. The implementation and operation of data-driven cloud services present new challenges and opportunities for future cloud infrastructures.

operational, and technical challenges faced when building and operating a new private-cloud environment. The architecture design, operation processes, and selection of the appropriate software components for a cloud environment depend on user needs, current virtualization environment, availability of skilled resources, and budget.

Our main recommendations are to completely integrate the new cloud environments into the existing data-center infrastructure and processes, and to automate the clouds' operation and maintenance processes to gain infrastructure efficiency and agility.

The software extensions and customizations described here are available for download at github.com/fasrc. 

REFERENCES

1. M. Nabi, M. Toeroe, and F. Khendek, "Availability in the Cloud," *J. Network and Computer Applications*, vol. 60, issue C, 2016, pp. 54–67.
2. R.S. Montero, R. Moreno-Vozmediano, and I.M. Llorente, "IaaS Cloud Architecture: From Virtualized Datacenters to Federated Cloud Infrastructures," *Computer*, vol. 45, no. 12, 2012, pp. 65–72.
3. I. Foster et al., "Virtual Infrastructure Management in Private and Hybrid Clouds," *IEEE Internet Computing*, vol. 13, no. 5, 2009, pp. 14–22.
4. S. Weil et al., "Ceph: A Scalable, High-Performance Distributed File System," *Proc. 7th Symp. Operating Systems Design and Implementation (OSDI 06)*, 2006, pp. 307–320.


JUSTIN RILEY is the team lead for Harvard University's Cloud & Software as Infrastructure Group. Contact him at justin_riley@harvard.edu.

JOHN NOSS is an infrastructure engineer at Harvard University. Contact him at noss@harvard.edu.

WES DILLINGHAM is an infrastructure engineer at Harvard University. Contact him at wes_dillingham@harvard.edu.

JAMES CUFF is the assistant dean and distinguished engineer for research computing at Harvard University. Contact him at james_cuff@harvard.edu.

IGNACIO M. LLORENTE is the director of the OpenNebula open-source project, a visiting scholar at Harvard University, and a full professor at Complutense University. Contact him at llorente@computer.org.

 ur private-cloud strategy can serve as a framework for understanding and successfully addressing the architectural,

 Read your subscriptions through the myCS publications portal at <http://mycs.computer.org>

