OPENSTACK

The Future of Virtualization: Modernizing Your Data Center with OpenStack

A VMware-to-OpenStack Migration Guide

Contents

1. Foreword
2. The Benefits of Moving to Open Source
3. Escape Proprietary Lock-In and Leverage
4. Unleash Innovation with OpenStack’s Open Design
5. Section 5
6. Section 6
7. Section 7
8. Section 8
9. Comparing VMware Features to OpenStack
10. Essential Functionality
11. Section 11
12. Section 12
13. Section 13
14. OpenStack Features that Aren’t in VMware
15. VMware Migration Framework
16. Economic Considerations
17. Enterprise-Level Support while Avoiding Vendor Lock-In
18. Automation
19. Migration Tools and Solutions
20. Networking Models
21. Storage
22. Section 22
23. Section 23
24. Section 24
25. Backups & Restore
26. Section 26
27. Section 27
28. Section 28
29. Section 29
30. Section 30
31. Section 31
32. Section 32
33. Section 33
34. Section 34

35. Section 35
36. Observability
37. Section 37
38. Section 38
39. Section 39
40. Resource Scheduling
41. Section 41
42. Section 42
43. Virtual Machine Conversion
44. Section 44
45. Section 45
46. Section 46
47. Section 47
48. Hardware Recommendations
49. General Considerations
50. Reusing What You Have
51. Reducing & Modernizing
52. Consolidation
53. Case Studies & Reference Architectures
54. Canonical
55. Huawei
56. Okestro
57. Mirantis
58. Wind River
59. ZTE
60. Non-Exhaustive Glossary of OpenStack Ecosystem Companies Offering Services
61. Authors

Foreword

November 2023 marked a turning point in the virtualization landscape. What began as just another high-profile acquisition — Broadcom's takeover of VMware — quickly escalated into a disruption that sent shock waves through the global IT community. The real impact came not from the deal itself, but from the abrupt and sweeping changes to VMware's licensing model. These so-called "licensing rug pulls" didn't just drive up costs for countless organizations, they shattered long-standing trust in a platform that had once dominated the virtualization market.

This wasn't an isolated incident. History has shown that sudden shifts in licensing models create widespread uncertainty and frustration, often leaving customers scrambling for alternatives. And in moments like these, open source solutions consistently emerge as the most viable, future-proof, and trusted path forward.

The virtualization landscape has evolved rapidly, and as businesses seize the opportunity to ditch vendor lockin and modernize their infrastructure with a cloud-native approach, many are looking beyond VMware. Migrating to OpenStack has emerged as a highly favored alternative to enhance flexibility, cut costs, and future-proof IT environments. Last year, the OpenInfra Foundation saw a unique opportunity to educate the market on how OpenStack is well positioned for organizations to regain trust while simultaneously modernizing their infrastructure. We formed an OpenInfra Member working group and worked with industry expert Steven J. Vaughan-Nichols to develop a high level VMware Migration White Paper.

Many were convinced, the next question was how? And how much?

Transformation doesn't happen overnight. A successful migration requires strategic planning, the right tools, and a deep understanding of technical and organizational considerations. This migration guide developed by the OpenInfra Member working group details how a migration can be done, including reference architectures and production case studies.

From assessing your current VMware footprint to implementing a future-ready, cloud-native OpenStack solution, this migration guide delivers the insights IT leaders and technical teams need to take control of their virtualization strategy and build a truly modern data center.

Migrating from VMware to OpenStack isn't just about making the switch, it's about making a move that is both seamless and strategic. Before you take the leap, you need a clear picture of your technology stack. What dependencies could slow you down? Which workloads are ready to move now, and which need extra planning?

This migration guide is your roadmap to success. We'll help you identify potential challenges upfront, strategize for any complex dependencies, and ensure a frictionless transition to an open, future-proof cloud. No surprises. No setbacks. Just a well-executed migration that puts you in control.

Open technologies fuel relentless innovation, empowering you to integrate cutting-edge tools without barriers. By tapping into the power of the global open source community, you gain

access to battle-tested software, expert insights, and continuous improvements—all at zero licensing cost.

The future of virtualization is open. Are you ready to make the move?

Jimmy McArthur, Director of Business Development, OpenInfra
Linux Foundation

The Benefits of Moving to Open Source

Escape Proprietary Lock-In and Leverage

Single vendor virtualization platforms like VMware come at a high cost. Expensive licensing fees, rigid contracts, and proprietary features designed to keep you locked in make it harder to innovate and scale on your terms. Worse yet, reliance on vendor-specific tools creates unnecessary complexity, making future migrations a costly headache.

With OpenStack, you can eliminate vendor lock-in and leverage a global ecosystem of interoperable, open source solutions. Broadcom's price model changes have significantly increased annual VMware operating costs—and when you're in a position where the vendor has this kind of leverage, this could easily happen again. The most reliable and complete way to make this whole class of problems disappear is to adopt open source software platforms like Linux and OpenStack.

By switching to OpenStack, you break free from these constraints and embrace the power of open source. No more restrictive licenses. No more forced upgrades. Just complete control over your infrastructure. OpenStack's open source cloud ecosystem empowers you to optimize costs, drive efficiency, and choose where and how to run your workloads—whether in-house or with one of the many vendorssupporting OpenStack worldwide. If your initial provider does not meet your demands, the interoperable nature of OpenStack allows you to seamlessly switch to another vendor with zero downtime.

The future of your infrastructure should be in your hands, not controlled by a vendor.

When you adopt a pure open source platform, no vendor can take away your operating environment. Whether you choose to go it alone or to work with a partner, open source allows you to completely own the your IT infrastructure.

Unleash Innovation with OpenStack's Open Design

Build Your Cloud, Your Way

Why conform to a vendor's roadmap when you can create your own? OpenStack's open design philosophy gives you unmatched architectural flexibility to build a cloud that fits your unique needs.

With OpenStack, your IT team has the freedom to customize every layer of your infrastructure, seamlessly integrate with cutting-edge technologies, and scale without limitations. Whether you need high-performance computing, edge deployments, or hybrid cloud solutions, OpenStack adapts to your business, not the other way around.

And you can forget one-size-fits-all. With OpenStack, you shape your cloud strategy on your terms and evolve at your own pace, without vendor restrictions holding you back.

Your cloud. Your rules.

Fortified Security On Your Terms

In a proprietary ecosystem, your security is at the mercy of a vendor's release schedule. With OpenStack's transparent, open development model, you're in control. Audit the code, implement security patches immediately, and contribute to improvements without waiting for vendor approval. This proactive approach eliminates blind spots, strengthens security, and ensures your infrastructure is always resilient against emerging threats.

Relentless Innovation Powered by a Global Community

Unlike closed platforms, where new features trickle down at a vendor's pace, OpenStack thrives on continuous, community-driven innovation. Thousands of enterprises, researchers, and developers across the globe collaborate to push boundaries, delivering cutting-edge advancements faster than any proprietary system. There is no stagnation, just constant evolution, ensuring your cloud is always ahead of the curve.

Seamless Integration & Future-Proof Agility

Vendor lock-in traps businesses in rigid ecosystems, making modernization an expensive challenge. OpenStack, built on open standards, integrates seamlessly with Kubernetes, Ceph, Ansible, and the latest cloud-native technologies. This level of interoperability gives you the freedom to adapt, scale, and evolve—without costly migrations or compatibility headaches.

By embracing OpenStack and the Four Opens—the guiding principles of the OpenInfra Foundation—your organization gains unparalleled freedom, cost efficiency, and a cloud strategy built for the future.

Comparing VMware Features to OpenStack

OpenStack and VMware share more similarities than one might initially assume. While both platforms offer many of the same core functionalities, they organize and present them using different terminologies and architectural frameworks. These differences in structure and naming can often exaggerate the perception of how distinct they truly are.

Most of the core compute, networking, and storage capabilities are almost identical across both platforms, reflecting their maturity and adherence to years of experience and best practices. However, their presentation and architecture are where they have the most differences:

- OpenStack technologies are more discrete—a collection of microservices composed from several projects (Nova, Neutron, Cinder, Horizon, etc.),
- VMware is more monolithic, with everything integrated into large stacks (vCenter, vCD).

Most perceived differences arise from terminology. VMware is heavily branded, with terms like vMotion, vSAN, etc. OpenStack either has named projects that do not necessarily indicate their functionality (Nova → Virtualization, Neutron → Networking, etc.) or uses more generic terminology for features ("Live Migration" in OpenStack vs "vMotion" in VMware). Fewer, but still noticeable, differences can be attributed to OpenStack being open source and VMware being a proprietary product suite. Once you get past these, it becomes evident that most of the same functionality is available regardless of the platform you choose.

Essential Functionality

Compute Comparison

		
OpenStack can provide multiple compute-style resource types: • VM: Nova • Bare Metal: Ironic • Containers: Zun • Can integrate with networking (Neutron) to be shared between compute types Virtualization allows for resource usage optimization: • CPU oversubscription (RAM oversubscription possible, but not often utilized) for VMs • "Nested" flavor definitions allow for effective bin-packing of VMs • VMs can be live-migrated between hypervisors Resource utilization is controlled by placement service: • Balances hypervisor usage for VMs		VMware is optimized to provide VM compute resources: • Oversubscription of CPUs and RAM is supported. • Placement of resources is managed dynamically (DRS). • Live migration (vMotion) is supported. VM constructs contain virtual hardware, and resources are explicitly described: • Virtual disks (VMDKs) • HW description (VMX) • "Live" in Datastores (VMFS) • VMs connect to virtual switches (vSwitch/DVS) and/or virtual networks (NSX). • VMs can be converted to templates for re-use. • VM state can be saved via snapshots (tied to VMDK, in the case of storage.

• Policies for placement can be pluggable • VM construct is abstract VM Instances are created from Images (Glance) according to specifications provided by "flavors": • VMs don't "live" anywhere by default. (Technically, they live on hypervisors, but this detail is generally not exposed to end-users.) • Attached storage is a separate construct (Cinder) but can be integrated. • VM NICs connect to "logical" networks (Neutron). VM functionality is extended by integration with other services (e.g., Manila for file sharing, Swift for Object Storage, Octavia for LBaaS with Neutron). • VM state can be saved in snapshots (which are saved to Glance). • VM templating is provided by a combination of Images (Glance) and flavor definition.		

Networking Comparison

		
OpenStack has two types of networking: • Provider Networks → L2 (Linux Bridge, OVS) • Virtual Networks → L2, L3 (OVN) Provider Networks: • L2 domains passed directly to OpenStack • Can be shared by tenants and/or push traffic up the network		**VMware has two types of networking:** • HW Analogue → vSwitch, Distributed vSwitch • SDN → NSX HW Analogue: • Basic connectivity, moving the access layer to the hypervisor • L2 (VLAN) segregation to integrate with "upper" network layers • VMs connected to "Ports" and "Port Groups" that can be tagged or native VLANs

Virtual Networks: • Can be done as tunnels (VXLAN, Geneve) • Can be integrated with Provider Networks • Can integrate with 3rd party tools either internally (Neutron) or externally (VXLAN, GRE) • Can also be provided by 3rd party deployments (e.g., Cisco ACI) • Networking functions (L2-L7) are provided in multiple ways. • Utilization of SDN (OVN) • Integration with Neutron (ML2, Security Groups) • Other services (FWaaS, LBaaS/Octavia, Ceilometer)		• Allows integration with HW appliances and "SDN-unaware" software appliances SDN: • Allows for L3 tunneling to create logical networks • Allows for services and functional endpoints to be integrated into the network fabric • Recently closed off to 3rd party partners Integration of network functions (L2-L7) is provided with NSX: • Includes higher-level functions: LBaaS, FWaaS, etc. • Built-in observability • Policy-based networking • Multi-site network connectivity (e.g., stretched L2)

Storage Comparison

		
VMs (and Bare Metal) have built-in storage that is size-defined by flavor definition: • This is either on block storage (boot from volume) or "ephemeral" storage, which is stored on the hypervisor locally. • Additional services (Glance, Cinder,, Swift, Manila) provide other storage for machine images, attaching additional block storage, object storage, or network attached filesystems. Each storage service provides its own repository and does not necessarily depend on another type of storage.		**Each VM has one or more virtual disks (VMDKs) associated with it.** Virtual disks can be either file-based or a thin frontend for a "real" device. Virtual disks are stored in a repository, either local to the hypervisor or over shared (SAN, NAS) storage. File-based virtual disks have snapshot state stored within the disk file: • Virtual disks can have multiple snapshots, although these snapshots cause the disk file(s) to grow over time. • Virtual disks can also be defined as Virtual Volumes (vVols), which can utilize

Images are used both as a template and as a method to efficiently store state for snapshots. Images can be imported/exported as block storage volumes for backup/ restore capabilities. Block storage can be used for additional VM storage and can be served by multiple backends: • Block storage can take advantage of backend capabilities if exported via the Cinder driver for that backend.		storage array features at the backend to provide additional capabilities (replication, local-copy, etc.).

OpenStack Features that Aren't in VMware

		
SSH key or user data injection OpenStack provides out-of-the-box SSH key and user data injection via the metadata API endpoint with the help of cloud-init. The user can configure this dynamically. **Concept of ephemeral data and base images** In OpenStack the recommendation is to use virtual machine images that rely on base images as ephemeral. This follows the cloud-native approach and enables CoW (Copy on Write), which provides significant benefits in boot speed and resource savings (i.e., no data copy). **Adjusting based on your needs** OpenStack and its components are open source and can be adjusted if necessary. For example, new capabilities can be added, or third-party tools can be integrated.		**SSH key or user data injection** This capability doesn't exist out of the box in VMware. It is technically possible to inject keys or user data via a template and cloud-init or script into the VM. **Persistent disks only** By definition, all disks in VMware are persistent. **Adjusting based on your needs** VMware is closed source and cannot be modified. All missing requirements need to be implemented by Broadcom.

VMware Migration Framework

Economic Considerations

Building a virtualization strategy requires balancing initial costs for software licenses, hardware upgrades, and skilled personnel. While virtualization can improve hardware utilization and reduce physical infrastructure needs, the savings depend on efficient resource management and scale. Long-term economic success also hinges on factors like licensing models, support costs, and the ability to adapt to evolving workloads and business needs.

OpenStack is open source software, with no software licensing costs or subscription costs required to use the software. Since its acquisition by Broadcom, VMware has restructured its pricing from an à la carte model to a more comprehensive subscription model. The two main pricing tiers (VCF and VVF) are now subscription priced by CPU core per year, with some features (e.g., vSAN) priced as an uplift on top of that. Also, while Broadcom has maintained some lower-priced offerings, they are typically bound by restrictions to limit their applicability and drive users to the main VCF and VVF tiers. Using OpenStack immediately provides relief from the substantial VMware licensing costs.

The cost structure of supporting OpenStack is also different. Only a nominal ability to file support cases is included with VMware, except at the highest tier (VCF). This can depend on the vendor from which VMware is acquired, as some VMware partners/vendors do overlay support. In most cases, all vendor-related OpenStack costs are based on the support of the hosts being managed. Various support models are available, but, in most cases, they cost less than VMware licensing alone, and those cost savings can become substantial when you add support or management to VMware software subscriptions.

Open source software is not only less expensive to operate, but the savings gained from migrating from proprietary platforms can yield immediate benefits when reinvested. The reduction in operating costs is significant enough to absorb migration costs and still have savings during the transition. Often, this can yield immediate operational savings that can be put back into your environment to modernize applications, update hardware, or level up staff knowledge.

Enterprise-Level Support while Avoiding Vendor Lock-In

One of the biggest concerns when switching from a proprietary solution like VMware to an open source-based platform is support availability and options. Fortunately, OpenStack offers a robust support ecosystem, ensuring reliability, continuous updates, and expert assistance from the community.

Organizations leveraging open source solutions like OpenStack can benefit from various enterprise-level support options, including Long-Term Support (LTS) Releases and Upgrade Considerations.

VMware provides a tightly integrated and vendor-controlled upgrade path, whereas OpenStack follows a six-month release cycle, which may seem daunting, but empowers organizations with flexibility to adapt their environments to timely trends and opportunities. In 2023, the

OpenStack community also introduced the Skip-Level Upgrade Release Process (SLURP)which allows organizations to upgrade less frequently while still benefiting from strategic updates. Many OpenStack vendors offer Long-Term Support (LTS) and Skip-Level Upgrade Release Process (SLURP) capabilities.

Even with all of these options, choosing open source doesn't mean you are taking the journey alone; many commercial resources are available to help with architectures, upgrades, operations, and training. These companies can act as a guide for information, training, architectures, best practices, and references that can enable organizations to become self-sufficient.

How the Global OpenStack Community & Ecosystem Can Support Your Organization

OpenStack boasts a strong, global community of contributors who continuously improve the platform. Organizations adopting OpenStack benefit from:

- Extensive, well-documented resources for setup, configuration, and troubleshooting.
- An active developer and user community providing real-time discussions, expert guidance, and shared best practices.
- Regular updates and security patches, driven by community contributions and the OpenStack Foundation.
- OpenStack's open source nature, which allows organizations to customize the platform to their needs, free from proprietary vendor constraints.

While OpenStack is open source, organizations could have access to enterprise-level support services from multiple vendors, offering:

- Professional services for deployment, maintenance, and troubleshooting.
- Security updates and long-term stability to ensure a reliable infrastructure.
- Managed OpenStack solutions for organizations needing fully supported environments.

Unlike VMware, where businesses rely solely on vendor-controlled support, OpenStack offers both enterprise-backed assistance and a strong, engaged community, providing flexibility, cost efficiency, and open innovation.

Automation

In today's fast-paced IT landscape, automation is no longer about luxury or convenience, it is about survival in a world where speed and efficiency define who wins. From the moment infrastructure is deployed, automation shapes how resources are provisioned, scaled, and managed. The ability to eliminate manual processes, optimize resource utilization, and ensure seamless scalability defines the efficiency of a cloud platform. The right automation strategy can mean the difference between seamless scalability and constant firefighting.

So, how do OpenStack and VMware compare when it comes to automation? OpenStack and VMware take very different approaches—one prioritizes flexibility and openness, the other reinforces vendor lock-in and costly add-ons.

Let's have a look at the key benefits of the ways to automate, which make OpenStack stand out in automation:

Open and flexible automation framework
With OpenStack's modular and API-driven design, teams can integrate powerful automation tools like Ansible, Terraform, Heat, Rockoon, and Kubernetes, without being locked into a single vendor's ecosystem.

The OpenStack Heat service supports reusing existing CloudFormation stack files written for AWS.

Cost-effective with no vendor lock-in
OpenStack's automation tools are open source, reducing dependency on expensive proprietary software. Organizations can build and customize their automation stack without incurring additional licensing fees, leading to significant cost savings.

Scalability and self-healing capabilities
OpenStack automation supports dynamic auto-scaling and self-healing mechanisms, ensuring workloads are distributed efficiently and recover automatically from failures. VMware offers automation but often requires additional paid solutions for similar functionality.

Infrastructure as Code (IaC) integration
OpenStack natively supports Infrastructure as Code (IaC) through tools like Heat and Terraform, enabling full infrastructure automation through declarative configuration files. VMware provides some IaC support but is less flexible due to its reliance on proprietary solutions.

Control and customization
OpenStack users have full control over automation logic, workflow design, and tool selection, allowing for tailor-made automation strategies. While OpenStack offers complete control over automation, VMware locks users into its ecosystem, requiring additional licenses and limiting customization options.

OpenStack provides a highly flexible, cost-effective, and vendor-neutral automation framework, making it a superior choice for organizations looking for scalability, control, and open integration. On the other hand, VMware, while offering automation, comes with higher costs, vendor lock-in, and limitations in adaptability. For companies that want agility, cost efficiency, and full control over their cloud automation, OpenStack is not just an option—it is a more future-proof solution.

Migration Tools and Solutions

Several approaches and tools can facilitate the migration process. These include both open source and custom migration solutions such as:

- Coriolisby Cloudbase Solutions
- CONTRABASS Legatoby Okestro

- FishOS MoveItfrom Sardina Systems
- MigrateKitfrom VEXXHOST, Inc. (open source)
- Mirantis Migration Service
- OS-MigrateAnsible collection (open source)
- virt-v2v(open source)
- ZConverter
- vJailbreak(open source)

Networking Models

OpenStack supports a variety of networking technologies that can be used to build resilient and scalable network architectures for users.

Self-service is at the heart of most OpenStack capabilities, and networking is no exception. By leveraging virtual routers and technologies such as VXLAN and GENEVE for network segmentation, users in projects can create and manage their own networks with little to no assistance from traditional network operations teams. In this model, users can provide direct connectivity to a virtual machine instance by assigning a Floating IP to that instance, resulting in a 1:1 (public-to-private) Static NAT. Users can also isolate virtual machine instances behind load balancers and limit access to bastion or jump hosts and automation.

OpenStack can also operate in a more traditional "network administrator"-driven model, where VLANs are pre-plumbed to hosts and virtual machine instances are configured to use one or more networks leveraging physical network device gateways, such as routers or firewalls. In this model, OpenStack can still provide DHCP services to users, but network and subnet creation and management are dependent on the physical network configuration.

Flexibility is a core concept of OpenStack with the network as well. Neutron, the network service, provides an abstraction layer for the SDN backend that can be chosen based on its capabilities. In this way, Neutron allows the creation of complex networking architectures and solutions like service chaining, BGP setups, or other capabilities depending on the SDN solution.

Storage

Ceph and OpenStack

Cephis the most commonly-deployed storage backend for OpenStack. Ceph is open source Software Defined Storage, leveraging commodity hardware to deliver enterprise-grade block, object, and file storage from a unified system. Ceph is tightly integrated with OpenStack storage services, offering ephemeral storage for Nova, image storage for Glance, persistent block storage for Cinder, S3-compatible object storage for Swift, and network attached storage (CephFS, NFS, CIFS) for Manila. Ceph is a mature, enterprise-grade storage backend which is proven at all scales, from small home labs up to multi-datacenter global cloud infrastructures.

Ceph's unique stand-out feature for OpenStack is that it provides Copy-on-Write thin-provisioning for VMs, which allows for very rapid VM instantiation and reduces storage space and network traffic.

Ceph and OpenStack together offer a complete suite of modern storage services and administration features, allowing organizations to quickly deploy a fully-featured on-prem cloud for their users.

Highlighted Ceph Features

Data protection: Ceph protects data with either replication or erasure coding, and uses smart algorithms to build the durable and highly available storage whilst considering real world failure domains. Replication is normally recommended for block and file storage, while erasure coding can be used for archive tiers and object storage. Both data protection types can coexist in the same cluster. In fall 2025, Ceph will unveil a new, highly optimized erasure coding backend, which will empower OpenStack users to further reduce costs without sacrificing performance.

Direct data access: In Ceph, clients like virtual machines or interactive workstations access the data directly, without any need for gateways that might add latency or throughput bottlenecks.

Autonomous self-healing: Ceph uses its knowledge of the cluster failure domains (e.g. hosts, racks, network switches, etc.) to distribute data intelligently to optimize durability and high availability. Whenever devices fail, Ceph re-replicates degraded objects to ensure that they are fully durable in a minimal amount of time.

Elasticity: Ceph clusters are highly elastic, growing or shrinking on demand without needing costly lift-and-shift data migrations. Hardware replacements can be carried out without any downtime or coordination with end-users.

Scale-out storage: Ceph is a horizontally scalable store – simply add more nodes and data will automatically rebalance itself to offer increased capacity and performance.

Multi-Cluster: Ceph supports large, multi-datacenter environments in several ways, including stretch clusters (1 cluster across 2+ data centers), replication (image/volume replication across regions), backup, and other approaches. All of this flexibility has been deployed in countless environments to ensure each organizations' business continuity and disaster recovery needs are met.

Ceph design considerations

Ceph is a highly flexible system – given that it will serve as the durable backbone of your OpenStack cloud for years to come, it is critical to ensure that your Ceph system is architected to fulfil your unique requirements in terms of feature compatibility, cost, performance, and growth.

The Ceph project maintains comprehensive documentation for newcomers who are just getting started as well as more experienced operators who are working to optimize Ceph or carry out regular maintenance.

The Ceph community is highly active and helpful for new and experienced users. Information regarding the community Slack channels, mailing lists, and events calendar are all available here.

The Ceph Foundation is a non-profit fund which supports the Ceph project through various promotional activities. Several of its members offer commercial support and professional services for Ceph. Find more information here.

Other Storage

In addition to Ceph, many storage devices are compatible with OpenStack persistent storage (Cinder). Cinder simultaneously supports multiple backends, which are controlled by the volume type given when the volume is created. For instance, Cinder could have a Ceph and a SAN backend. This allows the reuse of existing SAN devices as well as the addition of devices for specific purposes. Most SAN-style devices (iSCSI, FC, NVMeoF) have drivers for OpenStack. The feature set depends on the individual driver, but the majority of devices support the necessary features to support the Cinder feature set.

The existing SAN drivers do not support OpenStack Nova or Glance. Although it is possible to run Glance on Cinder, the current implementation has a number of drawbacks. In most cases, it makes sense to have a Ceph cluster for ephemeral and image storage if a SAN is used for persistent storage.

OpenStack documentationand the storage device manufacturer's manual provide more information on configuration and feature set.

Storage encryption

OpenStack Nova and Cinder can encrypt block storage directly on the compute node with the help of the secrets store, Barbican. If this feature is configured, data is encrypted directly when leaving the virtual machine and only decrypted upon arrival back in the VM, so the data is protected both in flight and at rest. It is possible to have encrypted and unencrypted volume types in the same cloud, so data that does not need to be encrypted can be stored unencrypted to minimize the amount of CPU consumption by the encryption algorithm.

Backups & Restore

Although cloud instances should generally be as stateless as possible—and best practice recommends backing up the application rather than the instance itself—the reality is often quite different. Users frequently treat cloud instances like traditional virtual machines, manually altering their state. Consequently, there is a clear need for VM-level backups, which we will analyze in the following sections.
Protecting hundreds or even thousands of virtual environments typically makes agent-based solutions unmanageable. Agents must be installed and maintained for each application you wish to protect. Of course, in certain scenarios—such as with databases—it may be necessary to deploy agents, but, in general, you should avoid an agent-based approach whenever possible.

An instance, like any other virtual environment, usually consists of two main components:

metadata and data. Metadata—i.e., the instance configuration (CPU, storage, networking, etc.)—is supplied by various OpenStack services, such as Cinder for storage definitions and Nova for compute resources. Backup solutions must always capture metadata alongside the data to ensure a consistent view of the instance; otherwise, you might discover later that a newly attached volume was never included in your backups.

The critical element is the data itself, which resides on OpenStack volumes. OpenStack supports several volume types, including temporary (ephemeral), Nova, and Cinder volumes. Ephemeral volumes can be safely excluded from backups, while Cinder volumes—which hold persistent application data—must always be protected. In rare cases, however, users may unknowingly store important data on Nova volumes, so it's wise to include those in your protection strategy when necessary.

Disk-attachment backup strategy

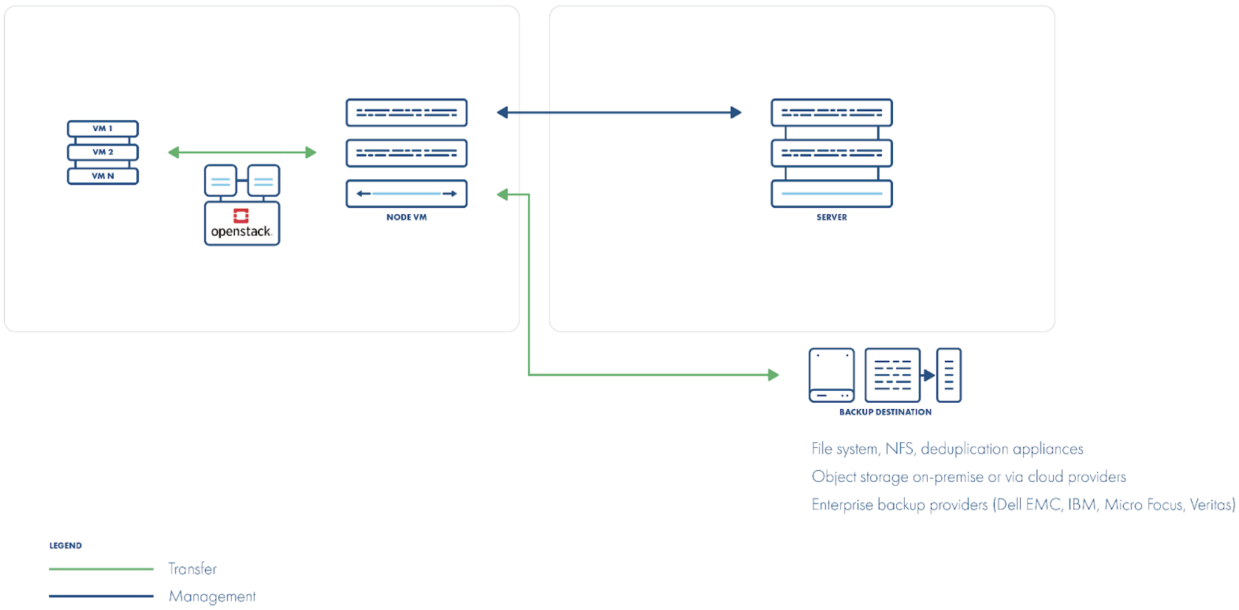
Depending on your storage configuration, you may need to adopt different strategies for backing up an instance. In the simplest—and most common—case, when you have only Cinder-based volumes and rely solely on OpenStack's core APIs, you can use the disk-attachment method.

This method assumes a helper (proxy) VM running your backup software, which reads data from snapshots of the protected volumes. Although the data is read directly from the underlying storage, OpenStack does not natively track which blocks have changed since the last backup. As a result, the backup application must scan the entire volume for modified blocks.

However, if you are using Ceph as your backend, you can leverage the Ceph RBD snapshot-difference API. In this scenario, you still mount volume snapshots on the proxy VM as with the disk-attachment method, but instead of reading every block, you call the RBD API to retrieve only the changes since the last snapshot, directly from the Ceph monitors. This requires network access to the monitors and retention of the previous snapshot so that Ceph can compute the delta.

Some vendors (including Storware) offer a checksum-based approach: they divide volumes into fixed-size regions (e.g., 64 MiB) and compare checksums between backups. If a region's checksum differs, the entire region is read. Although this adds processing time, it enables incremental backups on any Cinder-supported storage without relying on Ceph.

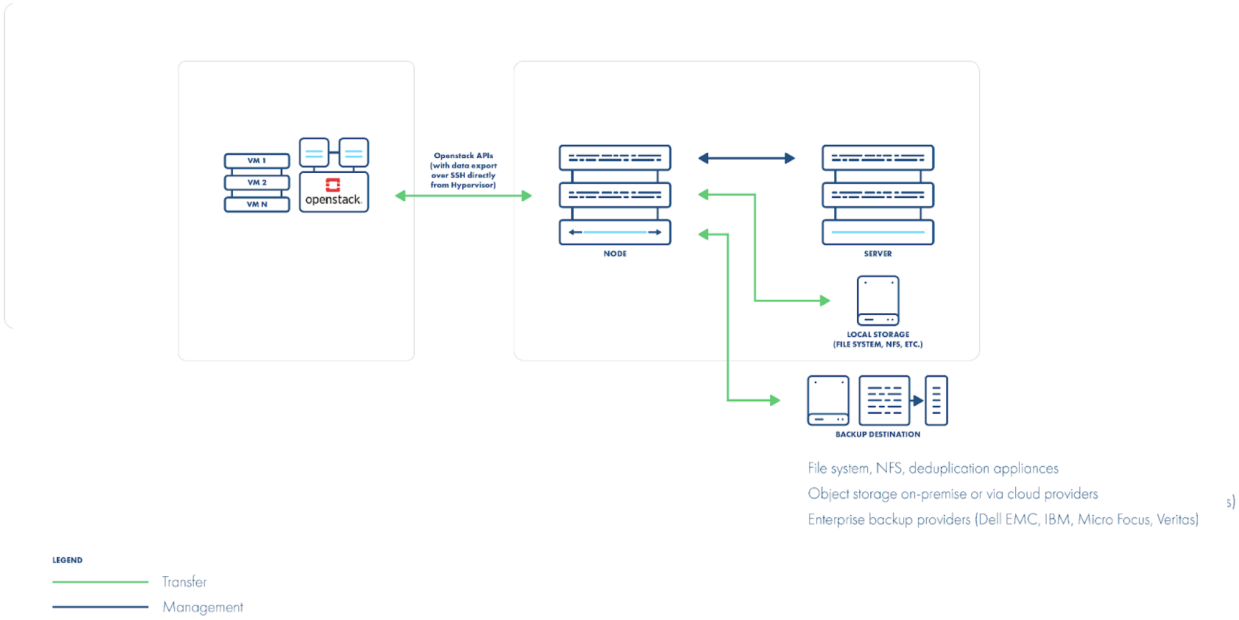
A still-simpler alternative is to perform full backups every time, and let a deduplication layer eliminate duplicate blocks, thus sidestepping the complexity of incremental backup altogether. In summary, the disk-attachment strategy covers a wide range of scenarios (including non-Ceph storage). Cinder provides a transparent abstraction over various disk arrays, and although OpenStack lacks built-in changed-block tracking (CBT), you can approximate it with snapshot-difference APIs, checksum scans, or by leveraging deduplication.



Direct transfer from host/storage backup strategy

One aspect the disk-attachment strategy cannot address is backing up Nova volumes. Although keeping application data on the same volume as the OS is not recommended, some users nonetheless do so. Nova volumes—provisioned for the guest OS—aren't exposed via the Cinder API and cannot be mounted like regular Cinder volumes. Instead, they typically live as QCOW2 file chains on the compute host (or on underlying storage) and require direct access for backup.

The direct-transfer strategy works at a different layer than disk-attachment: it speaks directly to the hypervisor (via libvirt) to locate and transfer the appropriate QCOW2 files. Snapshots, too, are taken directly on those QCOW2 images rather than through an API. Note, however, that VMs should not migrate between hosts during a backup, so QCOW2-delta snapshots must be retained only briefly.
Incremental backups are possible by using libvirt's dirty-block map feature, which tracks changed sectors on the VM disk without relying on snapshots. You can then export just those changed blocks (or perform a full export) via libvirt's built-in backup commands, with data flowing directly from the host. One caveat: for non-QCOW2 disk formats, dirty-block maps only track changes while the VM is running; a shutdown requires a full backup.



When your backend is Ceph RBD, you don't need dirty-block maps at all. You can snapshot the RBD volume directly in Ceph and stream it, which is much faster than reading QCOW2 files from the host. For incremental backups, you can either mount the RBD device (via RBD or RBD-NBD) and read only changed blocks, or leverage Ceph's native "diff" API to fetch deltas without mounting.
Some backup vendors may suggest installing agents or software on each hypervisor. We don't recommend this: vendor-specific host-level changes can break your OpenStack distribution or become unsupported after an upgrade.

Regardless of strategy, you must always fetch instance metadata via the OpenStack API. During recovery, you need to recreate an OpenStack instance, not just a libvirt VM. You may also need to import Nova volumes into Glance (as images) before you can boot the recovered instance.

Finally, consider how you handle VM images across data centers. If many VMs share the same base image, you don't want to restore that image for every VM (wasting storage). Your backup solution should detect whether the image already exists, or let you map to an alternate image ID if the primary data center's image isn't available.
As you've seen, this strategy requires no in-guest proxy: all data flows over the network from hosts or storage to your backup server. However, you do need network connectivity—and appropriate access credentials—to your hypervisors and storage systems.

Backup storage

Once instance data and metadata have been exported, they need to be stored securely. A common starting point is a traditional file system, which may offer deduplication, snapshots, or reflinks to build a synthetic backup repository.

At larger scales, however, you'll often need a more scalable backend—typically object storage. OpenStack provides a dedicated object-storage service called Swift, which also supports the S3 API. During implementation, you can choose which API to use for integration.

One important consideration is object size limits: Swift caps objects at 5 GB by default, so even modestly sized disks require object segmentation.

A less obvious advantage of object storage is that it can serve as a single backup target shared by all data movers. This enables instances to be backed up from one availability zone or data center and restored in another without the constraints of a traditional file system.

The downside is that restores may require rebuilding the entire backup chain, since synthetic merges aren't possible without a sophisticated data-chunking mechanism.

You should also consider object compression because object storage doesn't support sparse disks, and empty space inside your instances can unnecessarily consume significant capacity. Some vendors provide built-in compression and data segmentation for Swift, making instance size less of a concern.

Another key feature to evaluate is support for synthetic consolidation—the ability to present a fully merged "full" backup even when most runs are incremental. The naive approach—copying the entire full backup and overlaying deltas—consumes massive storage and, even with deduplication, forces you to rewrite the entire VM disk.

Vendors that support synthetic consolidation leverage storage-specific tricks (snapshots, reflinks, or similar) to create a virtual full copy instantly and then merge incremental changes without duplicating unchanged data.

Instant recovery

Instant recovery—which is ubiquitous in VMware environments but still emerging in OpenStack—allows you to mount and boot a VM directly from backup storage. This lets you rapidly verify data consistency and confirm that applications inside the restored instance actually start. When you have dozens or even hundreds of VMs to test, these on-the-fly restores become indispensable.

As the name implies, instant recovery must be truly "instant." That usually means your backup repository needs to present a fully merged image on demand—often via synthetic consolidation—so there's no merge delay at restore time.

In OpenStack, you can enable instant recovery by defining a custom Cinder volume type that points to your backup vendor's storage. When you launch a test VM, Cinder provisions its volumes from that vendor-backed pool, and the backup system dynamically serves the live data to the instance.

Slightly less obvious metadata that needs to be handled

In the VMware world, you typically only need to back up the VMX files, but in a true cloud environment, instance metadata is spread across multiple services. When restoring an instance, you must recreate its original compute-resource settings (the "flavor"). Because clouds standardize instance sizes, your recovery solution should simply select from the available flavors rather than inventing new ones.

Similarly, instances should be rebuilt from a centralized OS image. These images are static and contain no user data, so if you are restoring, say, ten identical instances, you should reference the existing Glance image (if it's still available) instead of re-uploading it ten times.

Networking adds another layer of complexity. Mapping virtual interfaces to the correct networks is straightforward, but security groups—OpenStack's equivalent of VMware port groups or firewall rules—must also be recreated or referenced if they already exist in the target environment.

Finally, consider SSH access keys. In OpenStack, you can't usually export private keys, so a robust backup-and-restore solution should let you specify an administrative key pair at recovery time. That way, you can regain access even if the original keys are lost.

Scalability

When building a large-scale environment, you must also consider the scalability of your backup solution. In scalable architectures, the component that typically needs to grow is the data mover, which handles the actual data transfer. To support larger deployments and run multiple backup jobs concurrently, you may need—or even be required—to deploy multiple data movers.
From a scalability perspective, it's important to check your vendor's limits on concurrent tasks or the number of backup threads each data mover can run. Fine-tuning these settings to match your infrastructure will help maximize throughput without overloading your systems.

When running concurrent backups, remember to scale your backup storage, too. A higher number of parallel streams—especially against scalable backends like Ceph—can quickly create a bottleneck on the receiving side.

You can mitigate temporary spikes by provisioning larger staging areas on each data mover to buffer data before it's sent to long-term storage. Alternatively, use scalable storage with high-bandwidth connectivity on the receiving end. Note that object storage usually prevents synthetic merges, but you can work around this by using a reflink-capable file system (for example, NFS 4.2+) as your backup target.

Finally, be aware that the disk-attachment method is constrained by the SCSI device limit per bus, often capping simultaneous exports at around 25 per node. In a large availability zone, this may become a bottleneck. However, some vendors let you deploy one data mover per

host, so in a 10-host zone, you could run ten data movers, each backing up only the VMs on its own host.

Horizon/Skyline integration

In a cloud environment, multi-tenancy is a fundamental consideration. OpenStack's Horizon dashboard enforces this by isolating each tenant within its own project scope, so users see only the resources belonging to their project.

Some vendors (including Storware) provide a Horizon plugin that adds a Backup tab to the dashboard. This plugin respects the user's current project context and exposes nearly all backup operations through an intermediary service that translates and forwards the appropriate API calls to the vendor's backend. This ensures that users can manage only the instances that exist—or previously existed, such as before accidental deletion—within their project.

This plugin lets users define backup policies and schedules and select the proper backup destination. The plugin's views and UI structure closely mirror the vendor's native interface, but are accessible directly within Horizon.

For managed service providers (MSPs), controlling which projects can see certain artifacts is essential. For example, you may want to restrict visibility of backup policies, backup destinations, or availability zones for restores. Some vendors let administrators configure project-level visibility settings for these items.

Backup quotas

Allowing end users to define their own backup policies—or to run frequent, overlapping backups—can place a significant load on your infrastructure. To control this, your backup solution should let administrators set quotas and assign them to individual projects.

Quotas can include both soft and hard limits for backup and restore operations. Hitting a soft limit generates an alert but still allows the job to proceed; exceeding a hard limit prevents any further backups or restores from starting.

You might restrict the number of backup or restore operations per instance—or per project—within a given time window. Alternatively, you may prefer to cap the total volume of data transferred rather than the count of operations.

Backup API

Although the Backup API capability is common in modern solutions, it wasn't always so. You'll eventually need access to your backup vendor's APIs to trigger backups, let the solution discover newly created instances, gather billing metrics, and more.

Check whether your vendor documents how their product can be automated. While a CLI can be handy for simple scripts, you should expect—and ideally require—a RESTful API to give you maximum flexibility in choosing integration technologies for your project.

Replication

Disaster recovery (DR) capabilities for your virtual machines (VMs) are essential for any mission-critical application. To provide seamless replication of a VM in OpenStack to another OpenStack cluster in a DR scenario, it is necessary to utilize third-party solutions such as Coriolis from Cloudbase Solutions. Coriolis does not require any agents to be installed on the VMs or the platforms, and it connects only to the necessary OpenStack services. Coriolis is built using a set of OpenStack components, such as Keystone, RabbitMQ, and Barbican, and can be fully integrated with your OpenStack environment.
In general, replicating data in an OpenStack environment can be achieved in several ways. The most basic method involves using the storage infrastructure itself, such as Ceph replication (or any other type of storage-level replication). However, the challenging part is connecting the replicated volumes to the instances that need to be created afterward.

Currently, few solutions support a typical per-instance replication scenario for OpenStack environments, but you can try to achieve it using existing building blocks. A basic approach would be to use backup software to perform full and incremental backups, store them (or replicate them) near the target OpenStack environment, and then periodically initiate instance recovery. The main drawback of this method is that, by default, a large amount of data must be transferred during each full restore. The advantage, however, is that even with currently available solutions, such as Storware, you can use its built-in mechanism for batch recovery (sometimes referred to as “recovery plans”).

In the near future, solutions for OpenStack will resemble SRM (Site Recovery Manager), where snapshot deltas are replicated to a secondary location, and only the deltas (incremental backups) are applied to the destination volumes. This would address the issue of large data transfers and long replication times and possibly increase RPO (Recovery Point Objective), if necessary.

Observability

In the world of cloud infrastructure, what you cannot see can hurt you. Whether it is performance slowdowns, unexpected failures, or security threats, having a clear view of your system’s health is critical.

OpenStack and VMware take fundamentally different approaches to observability, reflecting their overall design philosophies. OpenStack prioritizes flexibility, customization, and open standards, while VMware emphasizes integrated, vendor-managed solutions.

Let’s break it down. Here are a few key differences in the definition of observability in OpenStack VS VMware

Open Source vs. Proprietary Monitoring

OpenStack uses open source tools like Prometheus, Grafana, Ceilometer, and OpenTelemetry, giving organizations complete control over their monitoring setup and integrations. Conversely, VMware is a proprietary solution, with add-on components—e.g., vRealize Operations (vROps) and Aria Operations—requiring additional licenses. This approach forces users into multiple proprietary contracts, increasing the cost and making the solution less flexible for third-party tools. This approach leads to additional licenses, increasing the cost and making it less flexible for third-party tools.

Flexibility and Customization

OpenStack offers full control over telemetry data, dashboards, and alerts, making it easy to adapt monitoring to the organization's needs. While VMware provides built-in monitoring, customization is limited to VMware’s ecosystem.

Security and Compliance

OpenStack allows organizations to host monitoring tools on-premises, ensuring data sovereignty and compliance with security policies. VMware stores logs and metrics within VMware’s ecosystem, which may limit flexibility for regulatory compliance in certain industries.

For organizations seeking a cost-effective, flexible, and highly customizable monitoring solution, OpenStack offers vendor-neutral observability with open source tools and extensive integration possibilities. VMware, while providing a robust monitoring experience, locks users into its ecosystem with higher costs and limited customization options.

By choosing OpenStack, businesses can build a monitoring framework that aligns with their infrastructure needs, scales efficiently, and avoids vendor lock-in, ensuring long-term flexibility and control over their cloud environments.

Resource Scheduling

Using resources efficiently is a core tenet of cloud computing and the primary reason for one type of OpenStack resource scheduling, which focuses on virtual machines that are already running. Resource scheduling can also be used for the initial placement of virtual machines, enabling end users to securely distribute workload components to the proper locations.

Resource scheduling for the initial placement of virtual machines is based on availability zones, a filtering system that enables end users to choose virtual machine placement based on operator-defined options. The concept of availability zones is also used by most hyperscalers, including AWS, Azure, and GCP.

Each availability zone is a specific group of compute nodes, and the end user can choose if a VM should be placed in a particular zone for proper distribution. For example, some workload items should be placed as far as possible from each other (e.g., for a database cluster). OpenStack Nova schedules the virtual machine to the appropriate group of compute nodes, with the option to apply further conditions (filters), such as utilization of an appropriate node or other configured filters.

For virtual machines that are already running, OpenStack resource scheduling focuses on compute loads that might shift over time and lead to sub-optimal cloud utilization, such as having some compute nodes underutilized while others are under heavy load, or having nodes turned on but idle for periods of time.

Watcher

Watcheris an OpenStack service designed to optimize resources for OpenStack-based clouds. It collects metrics from a telemetry service and computes an action plan to optimize the cloud according to a goal chosen by the user. The action plan contains different actions to take, such as changing the power state of a compute node or live migrating virtual machines from one node to another.

Watcher datasources

The datasources are defined by Watcher as the telemetry services that provide the metrics to inform the state of the cluster. Multiple services are supported, like Prometheus, Gnocchi, Grafana, or Monasca, with the first two being the most commonly used. Watcher will typically get VM and host CPU, memory, or disk metrics, which allow it to design an optimization plan for the cloud.

Watcher modes of operation

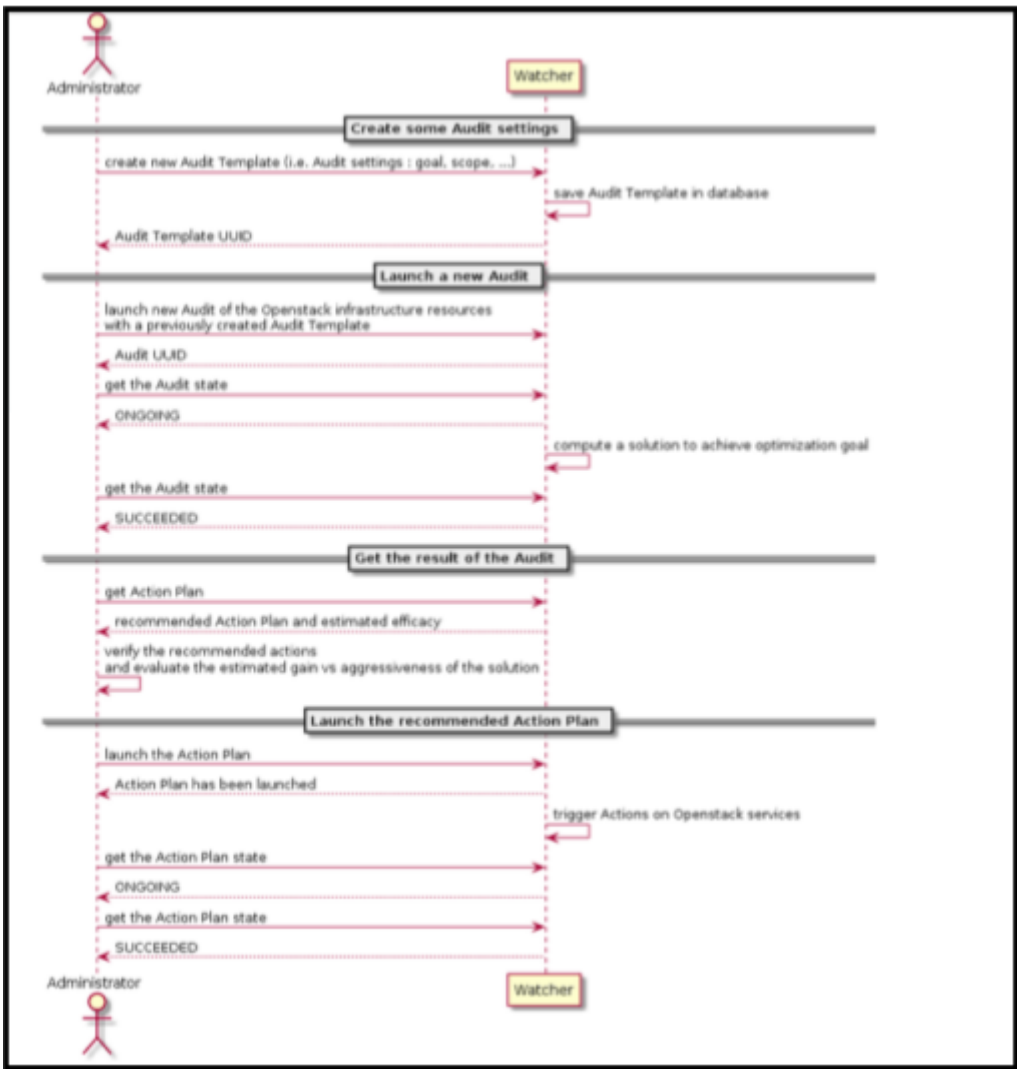
When using Watcher, the cloud administrator can opt for manual or automatic usage. In both modes, the user chooses a goal to optimize, and Watcher prepares a plan with changes to the cloud to bring it closer to the chosen goal. In manual mode, Watcher produces a plan that the user can either accept or not; if the plan is rejected, no change is made to the cloud. Otherwise, in the automatic mode, the proposed plan is applied to the cluster. Additionally, in both modes Watcher can be set to optimize continuously. If that setting is chosen, Watcher will propose new optimization plans in periodic intervals configured by the user, and the user can choose to accept the plans in manual mode or apply them automatically in automatic mode.

Example of typical workflow and overview of working strategies

Watcher provides a robust framework to achieve a wide range of cloud optimization goals. The user can select any of a number of supported strategies (algorithmic implementations) to help them find a solution for the given goal. This section will follow through some sample workflows that a cloud administrator might implement:

Distribute the workload evenly in the cluster (Workload balancing)
Consolidate instances on nodes (Node Resource Consolidation)

All workflows will run through the following main interactions between the Administrator and the Watcher system:



All workflows can be executed via the CLI or Horizon.

- Distribute the workload evenly in the cluster

  The Workload Balance Migration strategy allows the cloud Administrator to balance the cloud load by moving a workload whenever a server's CPU or RAM utilization percentage is higher than the specified threshold.

  The Administrator sets up a continuous audit to implement the Workload Balancing goal. The data source selected would be set up to monitor the server's metrics. If an audit detects that any server's CPU or RAM utilization exceeds the set threshold, an Action Plan with a recommended status becomes available. The Action Plan includes a Migrate Action to move the heavily loaded instance(s) to more evenly balance the server load. The Administrator selects to start the Action Plan. Once the utilization of the servers is back under the threshold, the Action Plan will be marked as succeeded.

- Consolidate Instance on Nodes

  The Node Resource Consolidation strategy checks the resource usage of compute nodes. If the used resources are less than the total, it will try to migrate servers to consolidate the use of compute nodes.

  The Administrator creates an Audit Template using the Node Resource Consolidation strategy for the goal of server consolidation. The Administrator then runs a oneshot Audit using the template created. If the Global Efficiency is not in line with the specified ratio, an Action Plan will be recommended. For example, if the Global efficacy shows Released_nodes_ratio: 50.00 %, this indicates that implementing this Action Plan will empty 50% of the Compute nodes.

  The Action list will show some pending migration actions. Once the Administrator starts the Action Plan, the server migrations will start, and the freed server will be re-enabled after the migrations are complete. The Administrator can then confirm that the servers have moved to the expected compute nodes and that all nodes are available for use.

Conclusion

Recent work to revitalize the Watcher project has shown that it can be a viable tool to help administrators optimize the utilization of infrastructure resources for OpenStack-based clouds. Going forward, Watcher could be further enhanced to expand the set of available strategies and incorporate relevant metric visualizations as action plans are implemented.

Dynamic Resource Balancer

Mirantis OpenStack for Kubernetes includes a Dynamic Resource Balancer (DRB) service, which eases the transition to OpenStack for VMware users who utilize automated resource scheduling and load balancing in everyday operations. For advanced OpenStack users, DRB also helps to maintain optimal performance and stability by preventing noisy neighbors and hot spots in an OpenStack cluster, which can cause latency or other issues.

The Dynamic Resource Balancer is an extensible framework that allows cloud operators to always ensure optimal workload placement in their cloud environment, without the need for manual recalibration. It continuously collects resource usage metrics for OpenStack nodes every 5 minutes and automatically redistributes workloads from nodes that have surpassed predefined, customizable load limits.

The Dynamic Resource Balancer consists of three main components: the collector, which gathers data about server usage; the scheduler, which makes scheduling decisions about workload placement; and the actuator, which live-migrates approved workloads to appropriate node(s). Cloud operators can limit workload redistribution only for specific availability zones or groups of compute nodes as needed. Additionally, the collector, scheduler, and actuator are Python plugins that cloud operators can customize with the most relevant data sources and decision criteria for their workloads and environments (e.g., to apply power supply metrics, use different observability tooling as the collector, or execute cold migrations).

Virtual Machine Conversion

VMware and OpenStack rely on different virtualization technologies and disk formats, making a conversion necessary before VMware virtual machines can run on an OpenStack hypervisor. VMware uses the VMDK (Virtual Machine Disk) format, while OpenStack works with QCOW2 or RAW disk formats.

Beyond disk format conversion, the guest operating system may require adjustments to boot correctly under KVM. This includes replacing VMware-specific drivers and kernel modules with virtio-compatible drivers and adapting to OpenStack's virtual hardware stack, which uses different network interfaces, controllers, and emulation models.

To perform this migration, several open source tools are available, such as virt-v2vand nbdkit.

Part of the libguestfs, virt-v2v is designed to convert virtual machines from foreign hypervisors like VMware ESXi into a format suitable for KVM/QEMU.

It performs:

- Disk format conversion (VMDK to QCOW2 or RAW)
- Guest OS reconfiguration: installing virtio drivers, updating boot settings, removing VMware dependencies
- Script injection: run custom scripts on first boot post-conversion for additional configuration or cleanup
- Cloud-init readiness, when desired, to allow OpenStack metadata injection at boot

Nbdkit is a flexible Network Block Device (NBD) server for serving disk images. It can act as a bridge between VMware storage and tools like virt-v2v, enabling efficient and streamed access to VMDK files without requiring full disk downloads beforehand.

Virt-v2v examples

Here are a few examples of how virt-v2v handles the virtual machine conversion from VMware to OpenStack.

1. Basic conversion via ssh to a local raw disk image, which could be uploaded into Glance at the end. Here, the ssh access needs to be configured in the ESXi / vCenter environment, and the path of the guest should be set according to your environment:

```
virt-v2v -i vmx -it ssh "ssh://root@192.168.122.8/vmfs/volumes/datastore1/
centos/centos.vmx" -o local -os /tmp/
```

2. Virt-v2v can also output directly to Qcow2 format:

```
virt-v2v -i vmx -it ssh "ssh://root@192.168.122.8/vmfs/volumes/datastore1/
centos/centos.vmx" -o local -os /tmp/ -of qcow2
```

3. Virt-v2v with direct upload to OpenStack Glance:

```
virt-v2v -i vmx -it ssh "ssh://root@192.168.122.8/vmfs/volumes/datastore1/
centos/centos.vmx" \
```

```
  -o glance -oc openstack \
  -on "centos-migrated-image" \
  -os "openstack"
```

4. Virt-v2v with a direct copy and conversion to Cinder volume, where server-id is an OpenStack instance from where the command should run and the Cinder volume attaches to it (needs to be run as root):

```
virt-v2v -ip passwd \
    -ic 'esx://root@192.168.122.8/Datacenter/192.168.122.1?no_verify=1' \
    -it vddk \
    -io vddk-libdir=/usr/lib/vmware-vix-disklib \
    -io vddk-thumbprint=XX:XX \
    -o openstack  -oo server-id=459b2b84-fcde-4ad9-9db2-2aa8c4c61445 \
    centosAll the detailed options can be found here: https://libguestfs.org/virt-v2v-input-vmware.1.html
```

Nbdkit examples

Here are some basic examples of using nbdkit to convert VMware guests to OpenStack:

Nbdkit with the vddk plugin allows you to consume the guest disk directly from the VMware environment, from the disk, a shutdown virtual machine, or from a live machine with a snapshot:

```
nbdkit --readonly vddk \
    libdir=/usr/lib/vmware-vix-disklib \
    server=vcenter \
    user=Administrator@vcenter \
    password=your_password \
    thumbPrint=XX:XX:XX:XX:XX:XX:XX:XX:XX:XX:XX:XX:XX:XX:XX:XX:XX:XX:XX:XX \
    vm=moref=vm-9 \
    snapshot=snapshot-1 \
    compression=skipz \
    transports=file:nbdssl:nbd \
    [datastore1] centos/centos.vmdkThen you can use nbdcopy to consume the network block device to a local disk:
nbdcopy nbd://localhost /tmp/local.disk --progressImportant: You can also directly write to a Cinder volume attached to an OpenStack instance.
```

Finally, use virt-v2v to convert the disk:
`virt-v2v-in-place -i disk /tmp/local.disk`Important: At this stage, you can benefit from all the options of virt-v2v, in particular the *–firstboot*, *–firstboot-command*, *–firstbootand-install*, and *–run*options to customize your disk during the conversion or the first boot of the virtual machine.

Ansible and automation

Where a virtual machine conversion is required and you will be using automation (possibly an Ansible script) as opposed to doing it manually, there is information that you need to have on hand to help the process. For example, you might gather information about the Guest and the VMware environment, such as the virtual machine paths, and also a specific value such as *moref*id or snapshot id.

There is for that purpose the GovmomiGolang package but also the Ansible collection for VMware, which provide valuable information for operating a virtual machine migration and conversion.

OS-Migrateand vJailbreakuses those libraries to automate and operate the entire migration and conversion process from VMware to OpenStack with very good performance and scalability.

Windows drivers

For Windows virtual machines, the virtio-winpackage provides all the required drivers for the conversion. Make sure you have the latest package version on your system in order to convert the latest Windows versions.

Hardware Recommendations

General Considerations

A wide range of hardware can be used for cloud computing, but the following principles guide good choices for hardware:

- **Failure Domain Design**: Cloud servers should be deployed so that physical failure domains are matched by logical failure domains (e.g., availability zones). This also applies to software-defined storage. OpenStack control planes should also be deployed across at least three failure domains to ensure continued operation in case of server failure.
- **Mid-Range**: Cloud servers should be mid-range between entry-level and top-of-the-line in CPU, memory, and storage configurations.
    - Entry-level servers make placing larger workloads difficult and may lead to wasting resources.
    - High-end servers are prone to network and storage bottlenecks, as many workloads compete for I/O. They also increase the blast radius in case of hardware failure, as a large number of workloads are impacted if a server fails.
- **Reduce Complexity**: Solutions that complicate the cloud configuration, maintenance, or lifecycle management should be avoided unless there are good reasons for the complexity. Examples are:
    - Too many cloud segments (availability zones and host aggregates) make placing workloads complicated and sometimes impossible, even though capacity is still available.

    - Compute nodes should ideally be of the same type to allow for easy replacement in case of failure and to avoid having multiple host aggregates due to different hardware types.

Reusing Existing Hardware

In a migration scenario, the OpenStack target cluster control plane is typically established first, along with a few compute and storage nodes. As the migration progresses, nodes in the source environment are freed up and often are redeployed in the target environment.

Many VMware environments consist of nodes suitable for OpenStack use. Most reasonably, recent x86_64 hardware can be used, though in some cases, memory augmentation, storage, or other resources may be required.

General recommendations for hardware configuration:

- **Flash storage**: Flash storage is recommended for control plane use in particular, but also for images and general block storage. NVMe is preferred due to the much wider performance envelope under pressure, but SATA SSDs can be considered too. Hard disks should only be used for archival storage types (backup) because they don't perform well under heavy parallel access, which is prevalent in most clouds. Many VMware environments are already all flash, but if the environment to be migrated is not, this would be a good time to replace magnetic storage with flash.
- **Memory configuration**: As many users share the same hardware, memory performance impacts general performance more than it would on most single-purpose nodes. In particular, all memory channels of a CPU should be populated. The number of memory channels from the CPU spec sheet must be multiplied by the number of CPUs to determine the minimum number of memory modules that should be used. Capacity should be adjusted by the size, not the quantity of memory modules. Existing servers typically can be easily modified to follow this recommendation.
- **Network and storage I/O**: Compute nodes with more powerful CPUs will concentrate more workloads on each node, which will compete for network and storage I/O. Bottlenecks can be mitigated by increasing the network ports in the server (additional NICs) or migrating to faster network infrastructure (e.g., 100GbE). As OpenStack installations typically use bonds for both performance and resilience, NIC ports should always be added in pairs or higher multiples.

Reducing & Modernizing

The question of whether older components should be reused/integrated boils down to three factors:

- **Economics**: Is the hardware still economically viable from the standpoint of power consumption, cooling requirements, and density in relation to the performance the hardware offers?
- **Reliability**: Is the hardware nearing the end of its life, which would make replacements more frequent and more challenging to obtain?

- **Density**: With rapidly increasing demand for compute, GPU, and storage, would upgrading existing hardware be cheaper than deploying additional racks or even data centers?

The advances of hardware capabilities over recent years have changed the game considerably. For example, 9 years ago, the fastest CPU on the market offered less than one-eighth of the core count and performance of the current commercially available market leader.

- CPU performance and thread count have skyrocketed. This allows for higher density in cloud environments, which reduces cost, power consumption, and cooling costs. CPUs cannot be upgraded across generations as the chipsets that connect them to the motherboard are CPU-generation specific.
- Memory can be increased by adding modules, but memory in older generation servers is much slower, and the CPUs have fewer memory channels than in current servers.
- Storage has become drastically faster as well as cheaper. At this point, hard disks are obsolete for anything other than purely archival/backup storage. Servers can typically be upgraded with SATA/SAS SSDs
- Network adapters can be upgraded by replacing the network interface cards.

As a rule, all servers older than 5 years should be thoroughly scrutinized for economic viability. The economic and performance calculations are customer-specific. In most cases, the best approach is to work with your cloud infrastructure vendor to determine the best economic solution for fulfilling the requirements for your target OpenStack environment.

Consolidation

Many VMware customers have a variety of small environments in production, which often grew over time instead of being designed at one point in time. Migrating to a cloud environment presents an opportunity for consolidation and proper architectural design of the target environment. The design phase should include a thorough hardware assessment.

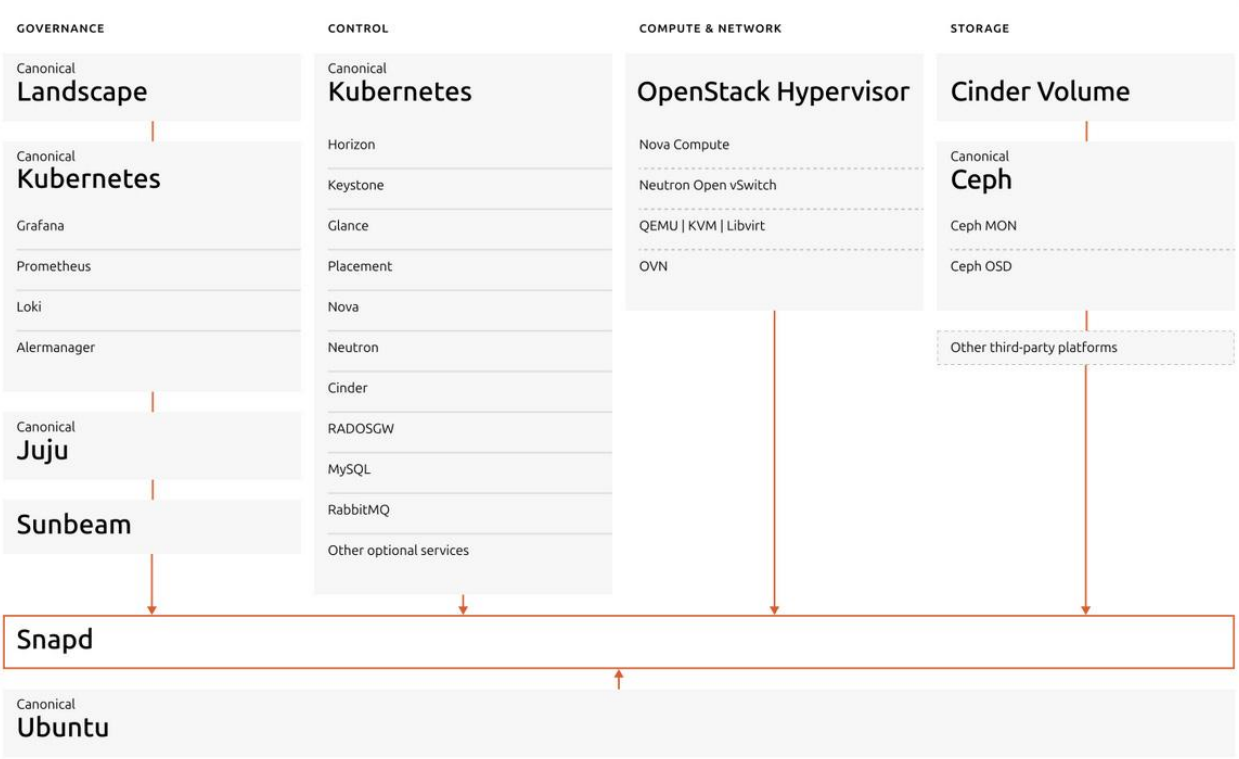
In 2024, the OpenInfra Foundationand the OpenStack VMware Migration Working Group collaborated with industry reporter Steven J. Vaughan-Nichols to develop the VMware Migration White Paper. This publication addressed the impact of Broadcom's acquisition of VMware and encouraged organizations looking to migrate to investigate OpenStack as an option.

Case Studies & Reference Architectures

Canonical

Reference Architecture

The Canonical OpenStack Reference Architectureleverages a modern, cloud-native design, where the control plane is orchestrated by Canonical Kubernetes and deployment is dramatically simplified by Sunbeam. This approach delivers a highly automated, scalable, and resilient private cloud, suitable for a wide range of enterprise and service provider workloads.



- Control Plane - Canonical Kubernetes-Managed: The OpenStack control plane is containerized and orchestrated using Canonical Kubernetes (Charmed Kubernetes or MicroK8s), providing robust automation, scalability, and lifecycle management. All major OpenStack control services—such as Keystone (identity), Nova (compute scheduling), Neutron (networking), Glance (image service), and Horizon (dashboard)—run as Kubernetes-managed workloads. This architecture enables seamless upgrades, high availability, and rapid scaling, while leveraging Kubernetes-native features for resilience and operational simplicity. The use of Kubernetes as the foundation ensures that OpenStack's API and RPC services, databases, and messaging layers are efficiently managed as microservices, with Charmed Operators (charms) handling deployment, scaling, and health checks.
- Sunbeam - Simplified Deployment and Operations: Sunbeam is Canonical's upstream OpenStack project designed to lower the barrier to entry and simplify both deployment and ongoing operations. Sunbeam uses full automation and high-level abstraction to enable even non-experts to bootstrap a production-grade OpenStack cloud in minutes. It leverages Kubernetes for container orchestration and Juju for charm-based automation, allowing users to deploy, configure, and scale OpenStack services just like any other cloud-native application. Sunbeam supports both small-scale and large-scale deployments, making OpenStack accessible for everything from single-node edge clouds to thousands of hypervisors in large data centers. With Sunbeam, post-deployment operations such as upgrades and scaling are also fully automated, minimizing operational overhead.

- Compute Nodes: Compute nodes run the Nova compute service and the virtualization stack (KVM, Libvirt/QEMU), hosting user workloads as virtual machines. These nodes can be scaled horizontally and may also run storage services in a hyperconverged setup. Software-defined networking (SDN) components such as OVN or OVS, and the Neutron agent, are also deployed on compute nodes to provide advanced networking features and metadata services for instances.
- Storage: The architecture supports both block storage (Cinder) and object storage (Swift or Ceph). Storage can be deployed on dedicated nodes or in a hyperconverged model with compute, allowing for flexible scaling and efficient resource utilization. Ceph is often used for unified block and object storage, providing high performance, durability, and scalability.
- Networking: Neutron delivers advanced SDN capabilities, integrating with OVN or OVS for automated network provisioning, segmentation, and security. Gateway nodes manage external connectivity, NAT, and floating IPs, while security groups and firewalls provide tenant-level network isolation and protection.
- Automation and Lifecycle Management: MAAS (Metal as a Service) automates hardware provisioning, turning bare metal into cloud-ready infrastructure. Juju, Canonical's model-driven operator framework, orchestrates the deployment, scaling, and lifecycle management of OpenStack services, enabling infrastructure-as-code and rapid operational changes.
- Monitoring and Management: Integrated monitoring tools and dashboards (such as Horizon and Landscape) provide real-time visibility into cloud health, performance, and security. This supports proactive maintenance through automated patching, centralized monitoring, compliance, and efficient operations.

Canonical's reference architecture, powered by Kubernetes-managed control plane and Sunbeam's automation, delivers a future-proof, highly automated OpenStack cloud. It lowers complexity, accelerates deployment, and ensures seamless scaling and upgrades, making OpenStack accessible for organizations of any size.

For full technical details, architectural diagrams, detailed references to the hardware and software architecture, please refer to the Canonical OpenStack Reference Architecture guide.

Case Study

Canonical has helped a wide variety of workloads migrate from VMware to their Canonical Openstack. Workloads migrated range from telecommunications to financial services to research and public services! Check out the list and details here in this Superuser post.

Huawei

Reference Architecture

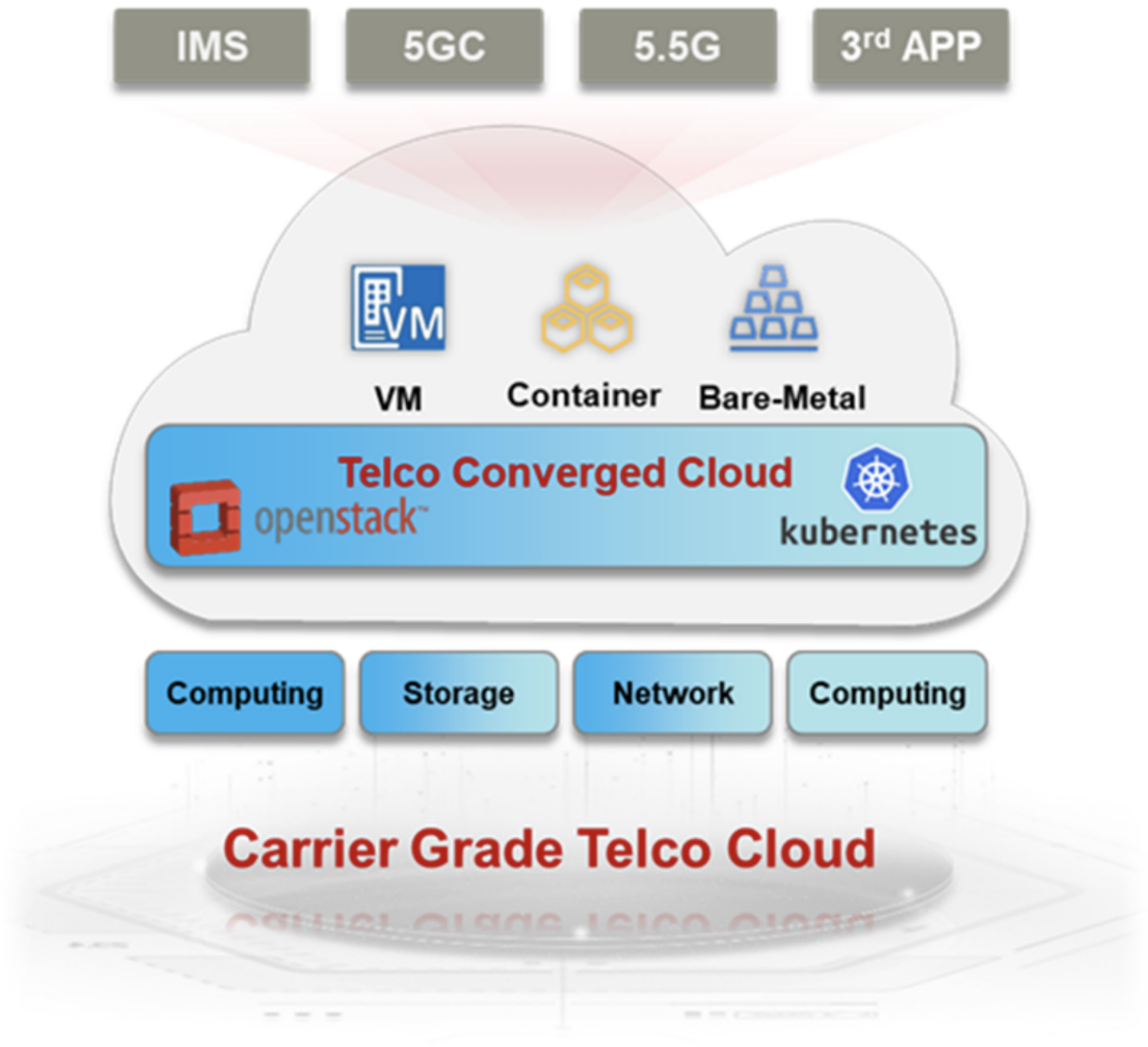
Huawei Telco cloud platform builds a converged dual-engine architecture based on OpenStack and Kubernetes. Working together with the OIF and CNCF community, Huawei telecom cloud platform uses OpenStack as the core cloud OS at the virtualization layer and enhances capabilities in telecom scenarios while contributing open source projects. Cooperate with community and telecom ecosystem partners, we together improve the performance, reliability, and openness of OpenStack and Kubernetes cloud platforms to meet strict requirements of

carrier-class services.

In terms of performance, the platform implements end-to-end customization and optimization by software and hardware collaboration to achieve optimal performance. Huawei has developed dedicated devices such as forwarding acceleration cards, network offload cards, and mesh cards at the hardware layer to meet the customization requirements of carrier-class services. The software layer is deeply enhanced based on the open-source framework and supports key capabilities such as dynamic huge page memory, core pinning, NUMA affinity scheduling, VF passthrough and etc., These effectively improve the overall computing and network performance.

In terms of reliability, the platform streamlines the collaborative scheduling mechanism of hardware, virtualization, network, and application layers to build an end-to-end high-reliability architecture and achieve carrier-class 99.999% availability. In addition, the system supports both VM HA and container HA, and provides extreme survival capabilities such as storage bypass, ensuring the continuity of mission-critical services in extreme scenarios.

In terms of openness, the platform architecture complies with the ETSI standards. It has good compatibility and continuous evolution capabilities, and effectively reduces the cost of interconnection with peripheral systems. The platform fully supports native open-source capabilities. Third-party applications can be deployed in the northbound direction and heterogeneous hardware are compatible in the southbound direction which provide carriers with greater flexibility and help carriers quickly to expand diversified services.



In terms of performance, the platform implements end-to-end customization and optimization by software and hardware collaboration to achieve optimal performance. Huawei has developed dedicated devices such as forwarding acceleration cards, network offload cards, and mesh cards at the hardware layer to meet the customization requirements of carrier-class services. The software layer is deeply enhanced based on the open-source framework and supports key capabilities such as dynamic huge page memory, core pinning, NUMA affinity scheduling, VF passthrough and etc., These effectively improve the overall computing and network performance.

In terms of reliability, the platform streamlines the collaborative scheduling mechanism of hardware, virtualization, network, and application layers to build an end-to-end high-reliability architecture and achieve carrier-class 99.999% availability. In addition, the system supports both VM HA and container HA, and provides extreme survival capabilities such as storage bypass, ensuring the continuity of mission-critical services in extreme scenarios.
At the delivery level, the automatic tool chain is introduced to automatically output key design documents, such as HLD, LLD, NSD, and VNFD, which reduces labor costs by 90%. In the

delivery phase, the platform supports pre-integration capabilities, improving the delivery efficiency by 50%. In addition, one-click VNF deployment and automatic ATP test cases are supported, which significantly reduces the manual workload during delivery and reduces the operation load by about 90%.

At the O&M layer, the platform provides a unified cross-layer visualized O&M portal to locate and demarcate faults on all links from hardware to applications, shortening the fault locating efficiency from days or weeks to minutes or days. In terms of upgrade, the platform supports compatibility between NFs and the N-2/N+2 version of the cloud platform decoupling. In addition, the system supports rolling upgrades in batches, ensuring service continuity and significantly improving system maintainability and scalability.

Case Study

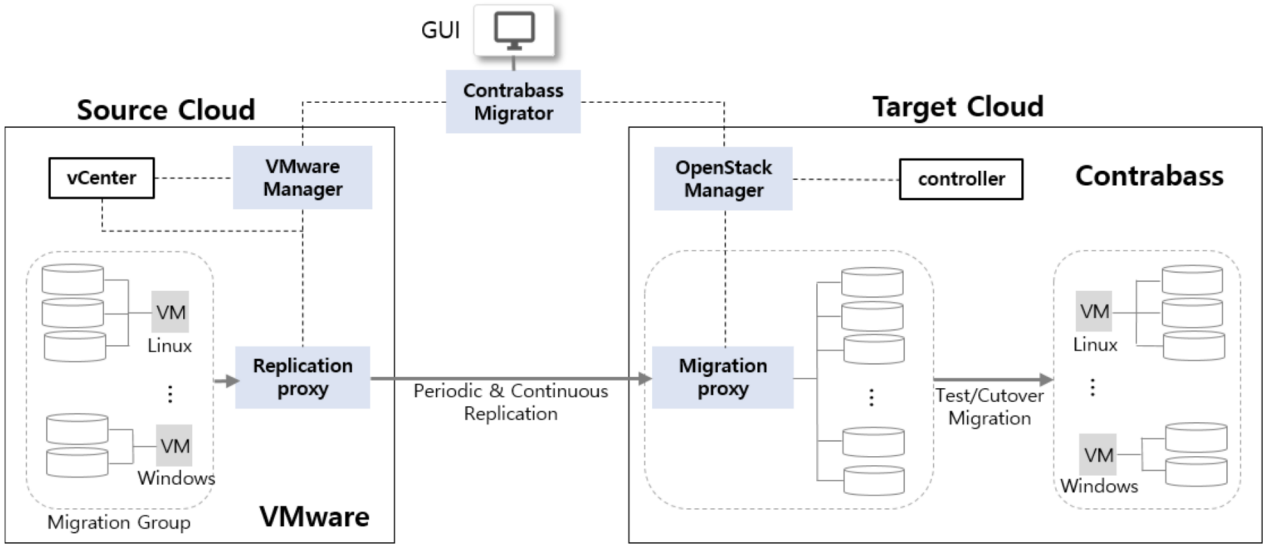
Huawei Telco Intelligent Converged Cloud (TICC) solution has demonstrated significant advantages in telecom cloud construction. The solution supports end-to-end automation from planning, deployment, to testing, greatly shortening the project rollout period (TTM). To find out more about how the TICC supports Vodafone as it continues to be in the forefront of telecom network cloudification, check out this case study!

Okestro

Reference Architecture

Contrabass Migration Service(CMS) is a highly automated lift-and-shift (rehosting) solution that allows you to migrate large numbers of virtual servers from VMware vSphere to OkestroContrabass without data loss or long cutover windows, using a web-based console with central management. Currently, CMS supports the migration of VMware vSphere virtual servers, but in the near future, it will also support migrations from physical servers, virtual machines, and cloud servers across a variety of hypervisors.

CMS periodically replicates source servers to Contrabass by taking snapshots and continuously replicates data using CMS's proprietary replication engine to provide a cutover window of only a few seconds. The replication engine provides real-time, asynchronous, file record-level replication for live migrations and operates at the OS level. Even without the replication engine, CMS ensures minimal downtime (a few minutes) and no data loss during migration.



CMS consists of five main components: Contrabass Migrator, VMware Manager, OpenStack Manager, Replication Proxy, and Migration Proxy. Optionally, the replication engine can be installed on each source server to achieve cutover windows of just a few seconds.

- **Contrabass Migrator**serves as the brain of CMS, coordinating and managing the entire migration process, including planning and execution.
- **VMware Manager**discovers and manages all relevant information about the source servers to be migrated, such as networks, IP addresses, and security groups.
- **OpenStack Manager**discovers and manages resources in the target cluster, including nodes for hosting the migrated servers and networks corresponding to those of the source environment. These resources can be configured via the web-based CMS console.
- **OpenStack Manager**also handles the full provisioning of resources such as network interfaces and firewalls required for launching instances on Okestro.
- **Replication Proxy**periodically or continuously replicates source servers to the Migration Proxy. The initial replication can take several hours to several days depending on the amount of data and the available bandwidth between the source and target clusters. CMS allows users to define specific replication windows to avoid impacting source system performance.
- **Migration Proxy**coordinates replication tasks, converts the server images, and adjusts the application stack, enabling the migrated servers to run smoothly on the target cluster.

Case Study

To learn more about the Contrabass Migrator tool and how Okestro leveraged it to support a government owned company in Korea, check out this Superuser Case Study!

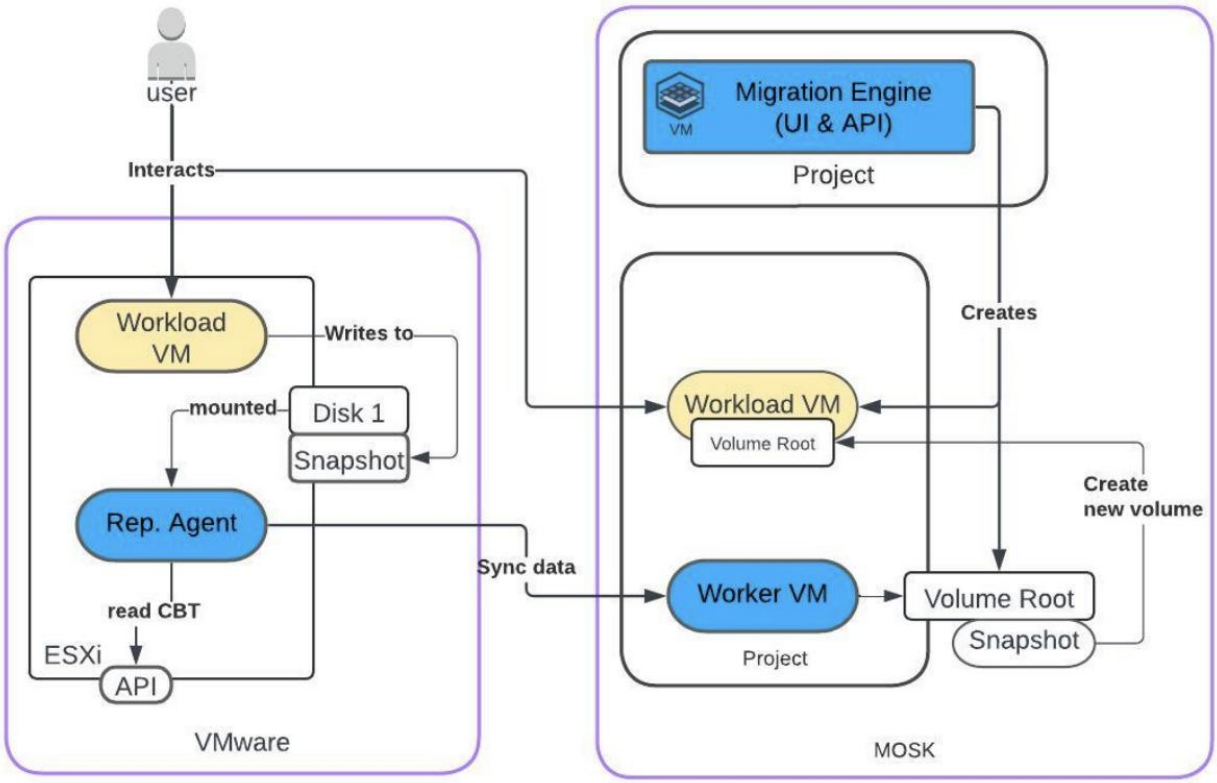
Mirantis

Workload Migration Architecture

The Mirantis workload migration service contains several components to automate the migration procedure from VMware to MOSK (OpenStack) with a typical Lift & Shift approach. The workloads are the most important entity so the tool provides capabilities to minimize the impact of availability during the cut-over. The virtual machine data is getting replicated to the target environment in the background and will not interrupt the running workload on the source side.

The initial replication is followed by booting up all machines on the target side to validate the workload functionality. After a successful validation the final cut over can be planned which means a short stop of the source machines and data transfer of the delta. It is possible to replicate the deltas constantly over a longer period of time. This allows execution of more complex tests on the target side while the source is still operating.

The migration can be executed in parallel to increase throughput and speed of the overall migration process.



Migration Engine- is deployed on the target side and manages the whole migration process. It provides an end-user friendly interface but also an API to extend automation or integrate in existing automation patterns.

Replication Agent- is running typically on each ESXi host as a virtual machine and scans the host of all available machines and general info of them. The rep. Agent is responsible for creating snapshots (enabling CBT) on source and initializing data transfer to the target worker machine.

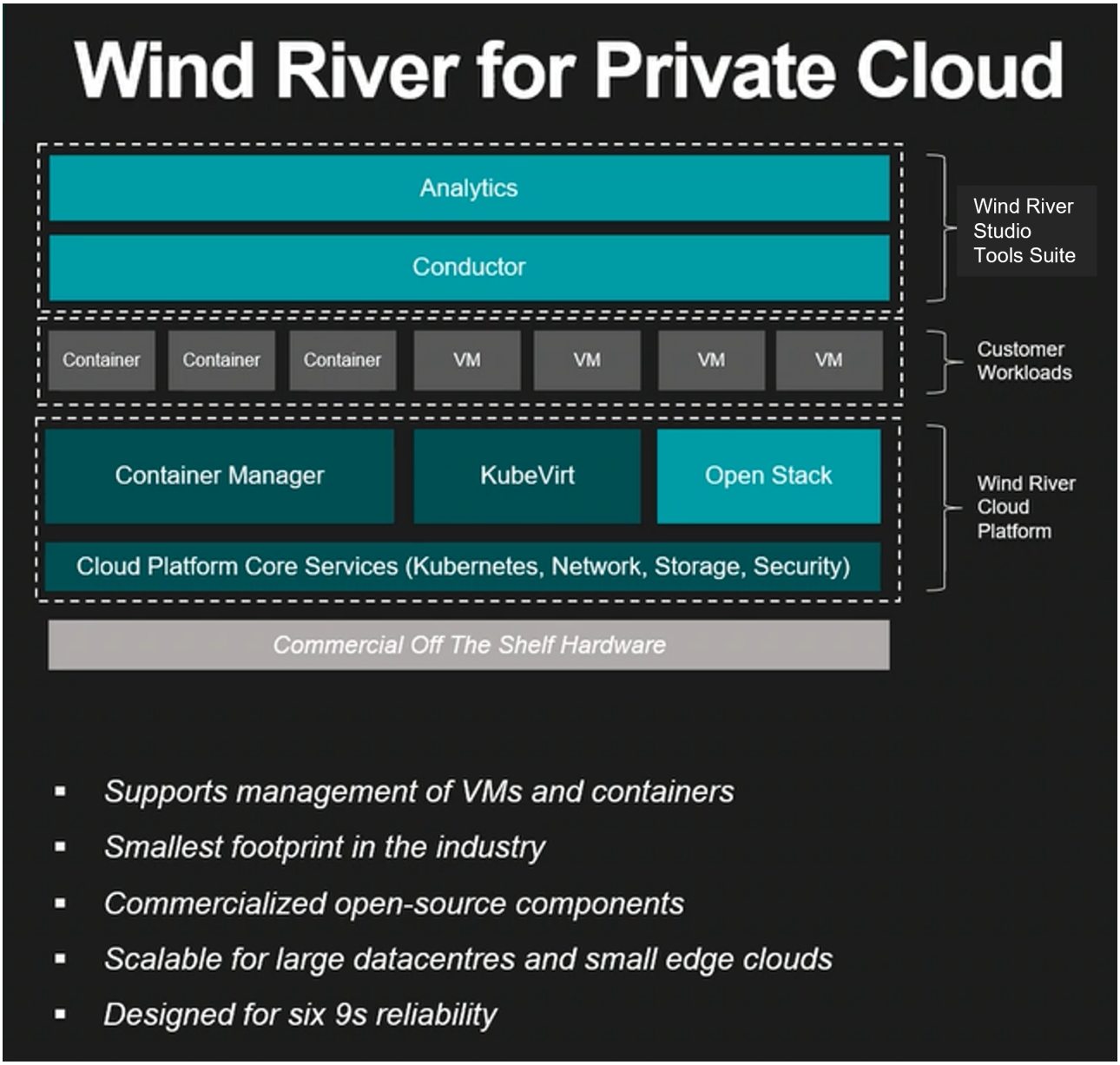
Worker VM- is spawned automatically in each target project temporarily for the time of the migration until the final cutover. The worker machine is responsible to attach newly created volumes (persistent storage) as a target for the source disks.

Case Study

With Mirantis k0rdent AI, Nebul, a leading European private and sovereign AI cloud provider based in the Netherlands, also has the operational efficiency and scalability to reach a broader market. The cloud service provider aims to expand to additional data centers across Europe, to help more customers safely join the AI revolution, while protecting their corporate data and ensuring compliance to a growing number of data privacy, security, and sovereignty regulations. To read more about this user case study, check out the full blog post here.

Wind River

Reference Architecture



Wind River has provided OpenStack-based solutions for many years and has since advanced its architecture to seamlessly integrate Kubernetes within a single, unified technology stack. This modern platform is powered by StarlingX, a leading open-source project purpose-built for edge and cloud-native deployments. With an integrated platform, automated orchestration, and built-in analytics, organizations can confidently optimize their private cloud environments while delivering low-latency performance for edge, near-edge, core and IT environments. Whether migrating from traditional VM workloads or enhancing existing private cloud operations, Wind River Cloud Platform delivers the reliability, flexibility, and cost efficiency enterprises need.

The migration of VMs and containers from VMware vSphere to the Wind River® Cloud Platform leverages OpenStack and Kubernetes to host both workload types within a single, integrated environment. This open, flexible, and cost-effective platform equips organizations with the tools needed to deploy and manage private cloud infrastructure globally, supporting both Operational Technology (OT) and Information Technology (IT) workloads across a geo-distributed network.

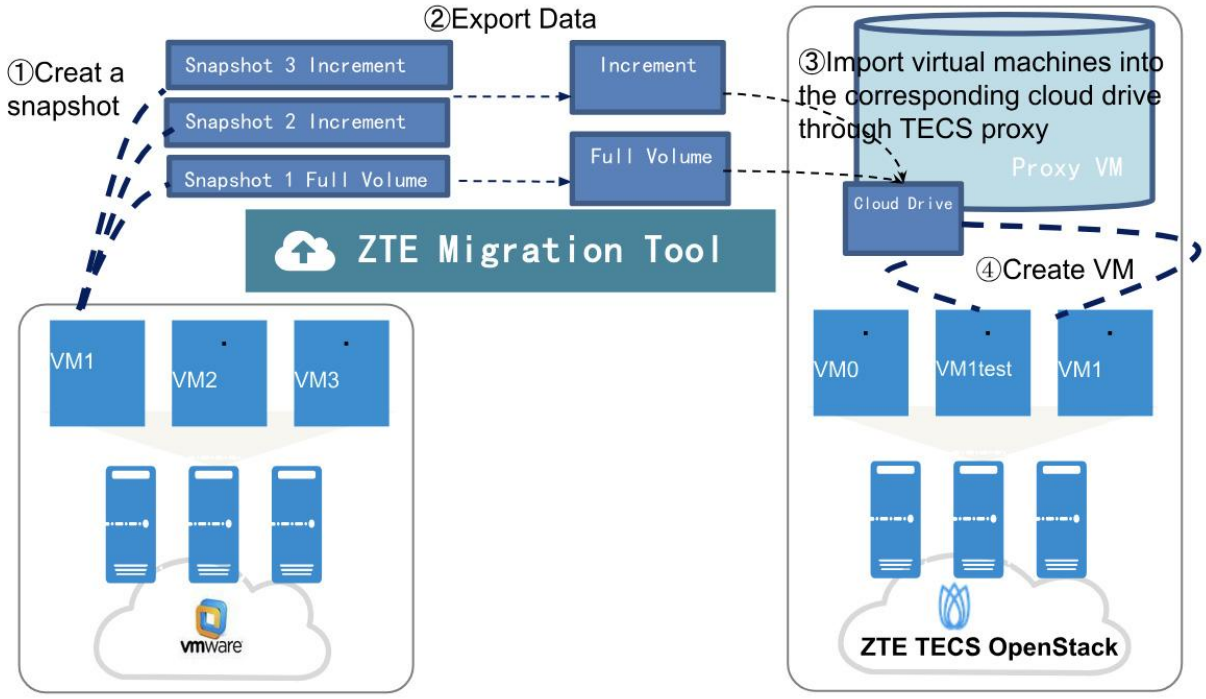
Case Study

Wind River has been helping organizations seamlessly migrate virtual machine (VM) and container workloads from VMware by leveraging its unique automation capabilities—even in some of the world's most complex and demanding cloud networks. Interested in learning more about their VMware alternative? Check out the post on Superuser!

ZTE

Overall Migration Process: "Four-Step Closed Loop" Ensures Smooth Transition

Taking a state-owned enterprise's test practice as an example, ZTE TECS migration tool constructs a secure and controllable end-to-end migration closed loop through four major phases: full replication, incremental synchronization, verification testing, and service cutover:



Phase 1: Full Data Replication
The tool automatically invokes VMware APIs to generate source VM snapshots and replicates complete data images to the target TECS cluster without affecting source operations, ensuring initial data consistency.

Phase 2: Incremental Data Synchronization
After full replication, periodic incremental synchronization is initiated. Differential snapshots

capture source data updates, dynamically aligning the target cluster with real-time source data to minimize cutover downtime.

Phase 3: Verification Testing
Post-synchronization, cloned test VMs are deployed in an isolated environment to simulate service scenarios. Under network isolation, operations teams validate data integrity and service availability to mitigate post-cutover anomalies.

Phase 4: Final Service Cutover
Minute-Level Downtime: After shutting down the source VM, a final incremental sync is performed, with the target automatically completing VM creation, network configuration, and driver adaptation.
Seamless Takeover: Post-service validation, traffic is redirected, compressing total downtime to minutes and ensuring critical business continuity.

Technical Highlights: Balancing Security and Efficiency

Agentless Mode Safeguards Data Security

Zero-Intrusion Design: Eliminates traditional agent-based approaches requiring VM plugins, directly leveraging VMware vCenter APIs for snapshots to prevent data tampering, meeting stringent financial industry integrity standards.

Resource Efficiency: Avoids agent-induced VM resource consumption, boosting migration performance.

API-Driven End-to-End Control

Flexible Scheduling: Supports customizable migration timing, incremental cycles, and cutover windows (e.g., off-peak/weekend), with checkpoint resumption to minimize business impact.

Rapid Rollback: One-click reversion to the source in case of validation failures, minimizing disruption and ensuring risk-controlled migration.

Customer Value: Enhanced Seamlessness and Autonomy

Seamless, Unnoticed Migration

API-driven scheduling ensures full migration completes during off-peak hours post-target readiness, while incremental syncs and cutovers are concentrated in low-traffic periods, delivering zero-interruption user experiences.

Core Technology Autonomy

Hardware Layer: Supports switching from Intel to domestic Hygon platforms.
Virtualization Layer: TECS platform achieves full-stack localization, reducing external dependencies and aligning with national innovation strategy requirements.

Non-Exhaustive Glossary of OpenStack Ecosystem Companies Offering Services

OpenStack Distros

- Atlancis
- AWcloud
- Canonical
- China Telecom
- Cloud & Heat
- Cloudbase Solutions
- Cloudification
- Coredge
- Debian
- EasyStack
- Ericsson
- Fujitsu
- H3C
- Huawei
- Inspur IEIT Systems
- Mirantis
- Okestro
- OSISM
- Red Hat
- Sardina Systems
- Taikun OCP
- Ultimum Technologies
- Whitestack
- Wind River
- ZConverter
- ZTE

Storage

- Ceph
- Dell Technologies
- NetApp
- Pure Storage
- Red Hat
- Scality
- Storware

Managed Kubernetes

- Taikun CloudWorks

OpenStack Public & Private Cloud Providers

- Atlancis
- Canonical
- China Mobile
- China Telecom
- China Unicom
- Cleura
- Cloudbase Solutions
- Cloud&Heat
- Cloudification
- DevStack
- Deutsche Telekom
- EasyStack
- Elastx
- Fairbanks
- Huawei
- Infomaniak
- Inspur IEIT Systems
- Mirantis
- Netways
- Okestro
- OpenMetal
- OVHcloud
- Planethoster
- Platform9
- Rackspace
- Red Hat
- Safespring
- Sardina Systems
- Sharktech
- Taikun OCP
- Ultimum Technologies
- Vexxhost
- ZConverter

Cost / Economics

- Canonical
- OpenMetal
- Sardina Systems
- Sharktech
- Spot by NetApp
- Vexxhost
- ZConverter

Observability

- Canonical
- Datadog
- Grafana Labs
- Hydrolix
- Mirantis
- Red Hat
- Backups & Recovery
- Cloudbase Solutions
- Commvault
- Hystax
- NetApp
- Storware
- Trilio
- Veeam
- Zadara

Support

- B1 Systems GmbH
- Canonical
- Cleura
- Cloudbase Solutions
- Cloudification
- Coredge
- Fairbanks
- Huawei
- Mirantis
- Okestro
- OpenMetal
- OSISM
- Red Hat
- Sardina Systems
- SUSE
- Vexxhost
- ZConverter

Authors

Marios Andreou, Mathieu Bultel, Ken Crandall, Ralph Dehner, Jennifer Fowler, Christian Huebner, Damian Karlson, Heiko Krämer, Natallia Kulazhenka, Ronelle Landy, Paweł Mączka, Jimmy McArthur, Joan Francesc Gilabert Navarro, Kendall Nelson, Dan van der Ster, Michelle Yakura, Christian Wolter