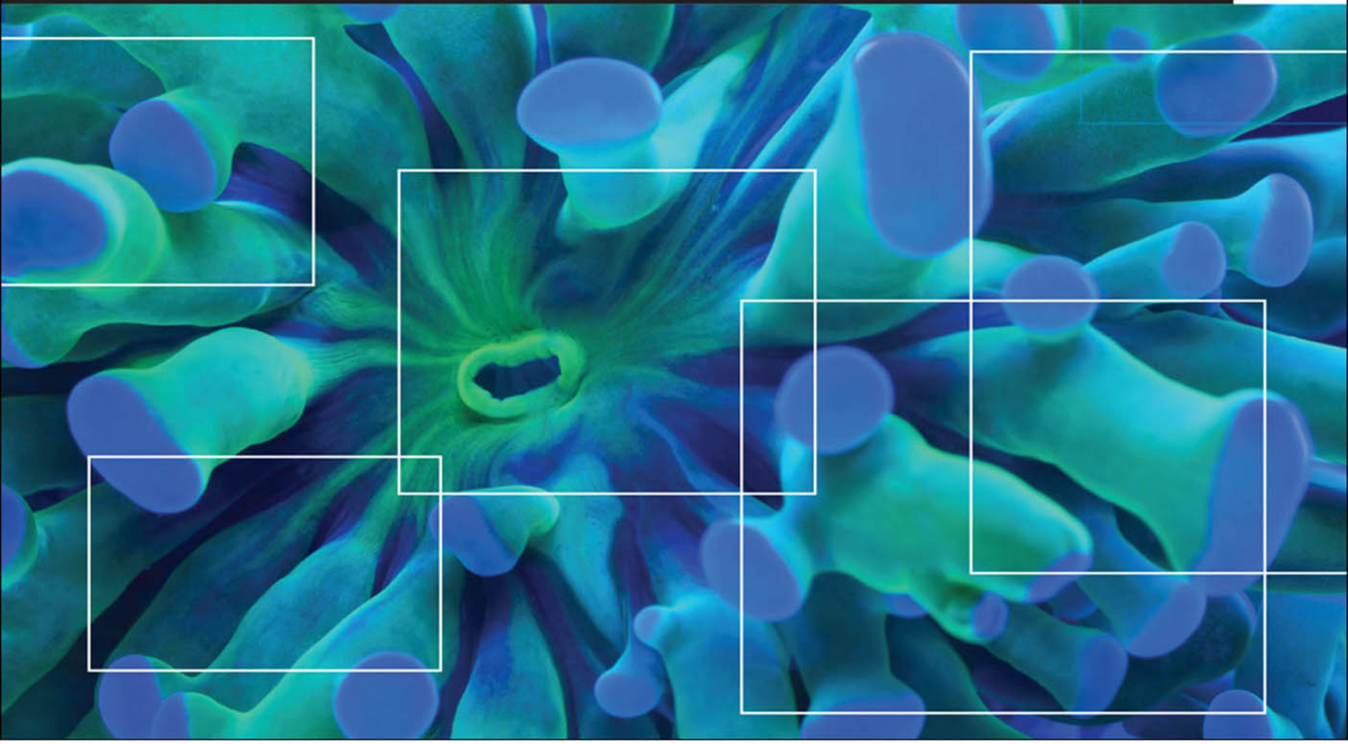
PRENTICE HALL

Oracle® Solaris 11 System Virtualization Essentials

Second Edition

Jeff Victor • Jeff Savit • Gary Combs • Bob Netherton



Oracle® Solaris 11 System Virtualization Essentials

Second Edition

This page intentionally left blank

Oracle® Solaris 11 System Virtualization Essentials

Second Edition

Jeff Victor, Jeff Savit,
Gary Combs, Bob Netherton

PRENTICE HALL

Boston • Columbus • Indianapolis • New York • San Francisco • Amsterdam • Cape Town
Dubai • London • Madrid • Milan • Munich • Paris • Montreal • Toronto • Delhi • Mexico City
São Paulo • Sydney • Hong Kong • Seoul • Singapore • Taipei • Tokyo

For information about buying this title in bulk quantities, or for special sales opportunities (which may include electronic versions; custom cover designs; and content particular to your business, training goals, marketing focus, or branding interests), please contact our corporate sales department at corpsales@pearsoned.com or (800) 382-3419.

For government sales inquiries, please contact governmentsales@pearsoned.com.

For questions about sales outside the U.S., please contact intlcs@pearson.com.

Visit us on the Web: informit.com/ph

Library of Congress Control Number: 2016959409

ISBN-13: 978-0-13-431087-9
ISBN-10: 0-13-431087-X

1 17

Jeff Victor dedicates this book to the memory of his sister, Diana Lyn Victor.

This page intentionally left blank

Contents

Foreword to the First Edition

I'm no longer sure when I first became hooked. Was it when I overheard a casual conversation about running a "test" copy of MVS in parallel with the real copy of MVS on a new 390 mainframe? Or was it the idea of Zarniwoop researching the *Hitchhiker's Guide to the Galaxy* in an electronically synthesized copy of the entire universe he kept in his office? Whatever the cause, I'm still addicted to virtual machine technology.

Fooling a whole stack of software to run correctly on a software simulation of the platform it was designed to run on has been a recurring interest in my career. Poring through the history of VM/370 as a graduate student, absorbing James Gosling's audacious idea of the Java VM, spending a few weeks building an experimental machine emulator to run SPARC applications on Solaris for PowerPC, the "aha!" moment when we realized how useful it would be if we arranged that a set of processes could behave as a little OS within an OS (the idea that became Solaris Zones), the first bring-up of OpenSolaris running as a paravirtualized guest on Xen—those are just a few of the highlights for me.

This book began as a project within Sun in mid-2009 during Oracle's acquisition of the company, so it both explores aspects of Sun's virtualization technology portfolio, and—now that the acquisition is complete—peers a little into 2010. Sun's unique position as a systems company allowed it to deliver a full set of integrated virtualization technologies. These solutions span the different trade-offs between maximizing utilization for efficiency and maximizing isolation for availability, while enabling the system to be managed at a large scale and up and down the layers of the systems architecture. Because that systems perspective informs everything we do, we have a wealth of solutions to match the diverse needs of modern enterprise architectures. Many of these tools are interoperable, enabling solutions that are otherwise impossible or impractical. Oracle's acquisition of Sun provides two further benefits to that portfolio: a secure future for these technologies and the exciting potential for integration with Oracle VM, Oracle Enterprise Manager, and the wealth of Oracle applications.

Here are some examples from the Sun portfolio. ZFS is a key storage virtualization technology at the core of the future of the Solaris operating system as well as the appliance products we build from Solaris technology today. Solaris networking virtualization technologies allow cutting-edge network hardware to be exploited and managed efficiently while providing a natural virtual network interface abstraction. For server virtualization, Solaris Zones (also known as Solaris Containers) have turned out to be very popular and very successful—a natural fit for the needs of many customers. The logical domains hypervisor is an extremely efficient design, and enables customers to get the most out of the tremendous throughput capability of SPARC CMT platforms. Our work with the Xen community enables a high-performance Solaris x64 guest for Oracle VM. For client virtualization, look no further than VirtualBox—for the laptop and desktop, both as a developer utility, and as a virtual appliance developer tool for the cloud. And it's not just a client technology: VirtualBox is the server component of Sun's virtual desktop infrastructure product, and it continues to grow more server-class features with every release. As well as infrastructure virtualization platforms, we have created infrastructure management software—Ops Center—intended to reduce the complexity that comes with using the new capabilities in large-scale deployments.

Virtual machines in one form or another have been around for a long time. Yet virtualization is such a fundamental idea that it remains associated with many developing fields. In the past decade, the runaway success of hypervisor-based virtualization on x64 platforms has largely been driven by the operational savings achieved by consolidating Microsoft Windows guests. But now this layer of the system architecture is just part of the way infrastructure is done—a new raft of capabilities can be built on top of it.

Recently we've seen the emergence of the Infrastructure as a Service (IaaS) style of cloud computing. Enabled by the combination of ever-increasing Internet connectivity and bandwidth, coupled with Moore's law about providing more and more computational power per dollar, users of an IaaS service send their entire software stacks to remote data centers. Virtualization decouples the software from the hardware to enable those data centers to be operated almost as a utility. This approach promises to revolutionize the fundamental economics across the IT industry. The capital expenditures currently devoted to under-utilized equipment can be shifted to pay-as-you-go operating expenses, both within large enterprises and between service providers and their customers.

This new layer of the systems architecture brings new opportunities and new problems to solve—in terms of security, observability, performance, networking, utilization, power management, migration, scheduling, manageability, and so on. While both industry and the academic research community are busily responding to many of those challenges, much remains to be done. The fundamentals remain important, and will continue to differentiate the various virtualization solutions in the marketplace.

This book is a deep exploration of virtualization products and technologies provided by or for Solaris, written by experienced practitioners in the art of delivering real solutions to data center problems. It provides a holistic view of virtualization, encompassing all of the different models used in the industry. That breadth itself is rare: No other organization has as complete a view of the entire range of system virtualization possibilities. A comprehensive background chapter leads neophytes into virtualization. Experienced data center architects will appreciate the individual chapters explaining the technologies and suggesting ways to use them to solve real problems—a critical resource in a rapidly changing world. I hope you find it as fascinating as I do!

Tim Marsland
Vice President and Fellow, Sun Microsystems, Inc.
Menlo Park
February 18, 2010

This page intentionally left blank

Preface

Computer virtualization has become a core component of the server industry; many organizations use virtualization in more than 75% of their servers. The portion of workloads running in virtual environments has increased in tandem with the maturity, number, and flexibility of virtualization options. Further, virtualization has become a required enabler of cloud computing.

Oracle® Solaris 11 System Virtualization Essentials presents the multiple technologies that the Oracle Solaris operating system uses to virtualize and consolidate computing resources, from hardware partitioning to virtual machines and hypervisors to operating system virtualization, commonly called "containers." The intent of *Oracle® Solaris 11 System Virtualization Essentials* is to discuss computer virtualization in general and to focus on those system virtualization technologies provided by, or that provide support to, the Oracle Solaris operating system. Oracle Solaris 11 supports a rich collection of virtualization technologies:

- Physical domains
- Oracle VM Server for SPARC (previously called Logical Domains)
- Oracle VM VirtualBox
- Oracle Solaris Zones (previously called Solaris Containers)

Virtualization offers a tremendous opportunity to add computing workloads while controlling operational costs and adding computing flexibility. For the system administrator, this new knowledge area requires skills with new technologies such as hypervisors, which create virtual machines on a single hardware machine, and containers (also known as zones), which create virtual operating systems running on a single complete operating system.

Oracle® Solaris 11 System Virtualization Essentials describes the factors that affect your choice of technologies. Along the way, it explains how to achieve the following goals:

- Use physical domains to maximize workload isolation on Oracle SPARC systems
- Use Oracle VM Server for SPARC to deploy different Oracle Solaris 11 environments on SPARC systems
- Use Oracle VM VirtualBox to develop and test software in heterogeneous environments
- Use Oracle Solaris Zones to maximize the efficiency and scalability of workloads
- Use Oracle Solaris Zones to migrate Solaris 10 workloads to new hardware systems
- Mix virtualization technologies so as to maximize workload density

Oracle® Solaris 11 System Virtualization Essentials contains nine chapters. Chapter 1 discusses system virtualization in general terms. This material includes the needs driving consolidation, the value and benefits of virtualization, and the most common types of computer virtualization. In addition, Chapter 1 also describes many of the concepts, features, and methods shared by many implementations of system virtualization. The concepts introduced in Chapter 1 are subsequently explored in much more detail in the other chapters.

Modern virtualization has been put to many varied uses. Chapter 2 introduces a few of those uses from a generic standpoint, tying benefits to features and providing simplified examples.

Chapters 3 through 6 hone in on Oracle's computer virtualization technologies that are directly related to the Oracle Solaris operating system. The large-scale deployment of virtual environments has created new system management challenges. In two different contexts, Chapter 7 reviews automation and management tools that can ease the pain of adopting virtualization solutions. Chapter 8 discusses the factors that should be considered when choosing a virtualization technology or combination of technologies, and suggests a process of analysis that can be used to choose a virtualization technology or combination of technologies. Assembling all of the pieces, Chapter 9 walks you through several real-world applications of those technologies. Finally, the Appendix offers a whirlwind tour of the history of virtualization.

Because this book focuses on system virtualization technologies, technologies and methods that do not virtualize a computer system are not discussed. These topics include, for example, storage virtualization and application virtualization.

Intended Audience

This book can benefit anyone who wants to learn more about Oracle Solaris 11. It is written to be particularly accessible to system administrators who are new to Solaris—people who are perhaps already serving as administrators of Linux, Windows, or other UNIX systems.

If you are not presently a practicing system administrator but want to become one, this book provides an excellent introduction to virtualization. In fact, most of the examples used in this book are suited to or can be adapted to small learning environments such as a home computer. Thus, even before you venture into corporate system administration or deploy Oracle Solaris 11 in your existing IT installation, this book will help you experiment in a small test environment.

Oracle® Solaris 11 System Virtualization Essentials is especially valuable to several specific audiences. A primary group is generalists who desire knowledge of the entire system virtualization space. The only assumed knowledge is general UNIX or Linux administrative experience. Another key audience is current and future data center staff who need an understanding of virtualization and use of such technologies in real-world situations.

- Data center architects will benefit from the broad coverage of virtualization models and technologies, enabling them to optimize system and network architectures that employ virtualization. The extensive coverage of resource controls can lead to better stability and more consistent performance of workloads in virtualized systems.
- Computer science students with UNIX or Linux experience will gain a holistic understanding of the history and current state of the system virtualization industry. The breadth of virtualization models discussed provides a framework for further discovery, and the real-world examples prepare students for data center careers.
- Technical support staff who troubleshoot virtualized systems will gain an introduction to system virtualization and interactions between virtualized systems. This background can shorten the time needed to diagnose problems, and enable personnel to readily distinguish between problems related to virtualization and ones that are independent of virtualization.

How to Use This Book

Readers who wish to learn about one specific Oracle Solaris virtualization technology should read Chapters 1 and 2, and the appropriate sections of Chapters 3 through 6, 7, and 9. If you would like to understand all of the virtualization technologies that use Oracle Solaris as a core component and determine how to choose among them, read all of the chapters in this book.

If you already understand virtualization but want to learn about virtualization using Oracle Solaris, you should skim through Chapter 1 to understand the context of the rest of the book as well as the definitions of terms used throughout the book, and then read Chapter 2 and the introductory sections of Chapters 3 through 6. Chapters 8 and 9 will also be especially useful.

If you are implementing virtualization technologies on many systems, you should read Chapter 9 to understand the unique problems that must be addressed as part of this work and to identify software that can significantly reduce the complexity of large virtualization farms.

Register your copy of *Oracle® Solaris 11 System Virtualization Essentials, Second Edition,* at informit.com for convenient access to downloads, updates, and corrections as they become available. To start the registration process, go to informit.com/register and log in or create an account. Enter the product ISBN (9780134310879) and click Submit. Once the process is complete, you will find any available bonus content under "Registered Products."

Acknowledgments

The authors would like to thank Detlef Drewanz and Doug Schwabauer for writing the two sections of Chapter 7 in the second edition. Bill Calkins entered late in the game to update the chapter dedicated to VirtualBox. The authors appreciate these valuable contributions.

The authors also appreciate the expertise and hard work of Prentice Hall executive editor Greg Doench, as well as freelance project manager Rachel Paul and copy editor Jill Hobbs.

Jeff Victor would like to thank his wife Cara and their daughter Kathryn for their patience and support during this endeavor. He also thanks John Fowler, Markus Flierl, and Dawit Bereket for their support during the production of the second edition.

Jeff Savit thanks his wife Vicky for her support during this effort. He also thanks Honglin Su, John Falkenthal, and the Oracle VM Server for SPARC organization for creating and growing this powerful virtualization platform.

This page intentionally left blank

About the Authors

This book is made possible by the hard work and expertise of the following contributing authors.

Jeff Victor is the principal author of *Oracle® Solaris 11 System Virtualization Essentials* and a Principal Sales Engineer at Oracle Corporation. Prior to joining Oracle, Jeff was a Principal Engineer at Sun Microsystems. He leverages his expertise in computer virtualization, operating systems, and network architecture to help organizations run more efficiently. He is a regular author, contributor, and speaker at corporate and industry events. His blog can be found at http://blogs.oracle.com/JeffV. Jeff received a bachelor of science degree in computer science from Rensselaer Polytechnic Institute. In his spare time, he travels with his family, leads an automated wildlife photography project, and investigates STEM topics with his daughter. Jeff lives in New York with his wife and daughter.

Jeff Savit is a product manager in the Oracle VM Server organization and specializes in operating systems, virtualization, and performance. Prior to Oracle, he was a Principal Field Technologist at Sun. Previously he was a Vice President at Merrill Lynch, with roles in systems programming, software development, market data, and web applications.

Gary Combs is a Sales Performance Designer in the Oracle Sales and Partner Academy at Oracle Corporation. He specializes in creating field sales training programs covering SPARC servers, and all Engineered Systems. Prior to joining Oracle, Gary was with Sun Microsystems. He has more than 15 years of direct sales support experience as a systems engineer, and more than 15 years of experience in marketing positions in product management, product definition, and technical marketing.

Bob Netherton is a Master Principal Sales Consultant at Oracle Corporation specializing in Oracle Solaris, virtualization, open-source software, and Engineered Systems. Prior to joining Oracle, Bob was a Principal Field Technologist for Sun Microsystems, and was one of the architects and content developers of the Solaris Boot Camp and Deep Dive seminar series. In addition, he has developed several best practices guides for Solaris as well as an advanced Solaris training curriculum. Bob received a bachelor of science degree in applied mathematics from the University of Missouri, and he is a regular blogger on Solaris, virtualization, and open-source technologies.

7

Automating Virtualization

Early computers were expensive, prompting their owners to squeeze all possible value out of them. This drive led to the introduction of time-share operating systems, on which many workloads may run at the same time. As per-unit cost dropped, single-user, single-workload operating systems became popular, but their adoption created the mindset of "one workload per computer," even on servers. The result was an explosion of under-utilized servers. The high costs of maintaining so many servers led to the widespread embrace of virtualization, with the goal of reducing the quantity of servers owned by organizations. Consolidation via virtualization may have reduced a company's hardware acquisition costs, but it did nothing to improve the organization's maintenance costs. Ultimately, managing VEs one at a time is no easier than managing one server at a time.

Many virtualization management tools exist on the market that can facilitate the process of automating virtualization. This chapter discusses two of them: Oracle Enterprise Manager Ops Center and OpenStack.

7.1 Oracle Enterprise Manager Ops Center

Oracle Enterprise Manager Ops Center 12c is part of the broader Oracle Enterprise Manager product. Whereas Enterprise Manager Cloud Control focuses on the higher end of the stack (i.e., database, middleware, and applications), Ops Center addresses the lower end (i.e., storage, operating systems, hardware, and virtualization).

Ops Center is designed for full life-cycle management of the infrastructure layer, which includes both Oracle hardware and operating systems. From a hardware perspective, it is capable of functions such as the following:

- Discovery of new and existing hardware
- Upgrading server firmware
- Installing the "bare metal" operating system
- Monitoring hardware components and opening service requests automatically if a hardware fault occurs
- Providing console access to the system
- Other management actions such as power-off/on, set locator lights, and others

Paramount in Ops Center's functionality portfolio is managing the two primary virtualization technologies: Oracle Solaris Zones and Kernel Zones, and Oracle VM Server for SPARC. Provisioning virtual environments (VEs) including those types includes performing any required preparation of the hardware and operating system.

7.1.1 Architecture

The architecture of Ops Center consists of three main sections:

- **Enterprise Controller:** The main server component of Ops Center. The enterprise controller delivers the user interface and stores the enterprise-wide configuration information. An organization that uses Ops Center will have at least one enterprise controller system that provides communication back to Oracle for service requests, automated patch and firmware downloads, contract validation, and other activities. However, many disaster recovery sites include their own enterprise controller so that they can continue operations management, if needed, during service outages that affect the rest of the system.
- **Proxy Controller:** The component that communicates to the managed assets, including hardware assets, operating system assets, storage assets, virtualized assets, and others. If all of the systems being managed by Ops Center are in one data center, only one proxy controller is needed, and it can run in the same server as the enterprise controller. Alternatively, you can install multiple proxy controllers per enterprise controller. Standard

configurations use one or more proxy controllers per data center, to expand the reach of the Ops Center environment to other data centers, networks, or DMZs.

- **Agent:** A proxy controller typically manages deployed software components via a software agent installed on the system. When an agent is not appropriate, an operating system can be managed without one. The Ops Center agent supports Solaris 8, 9, 10, and 11.

Figure 7.1 depicts the Ops Center architecture.

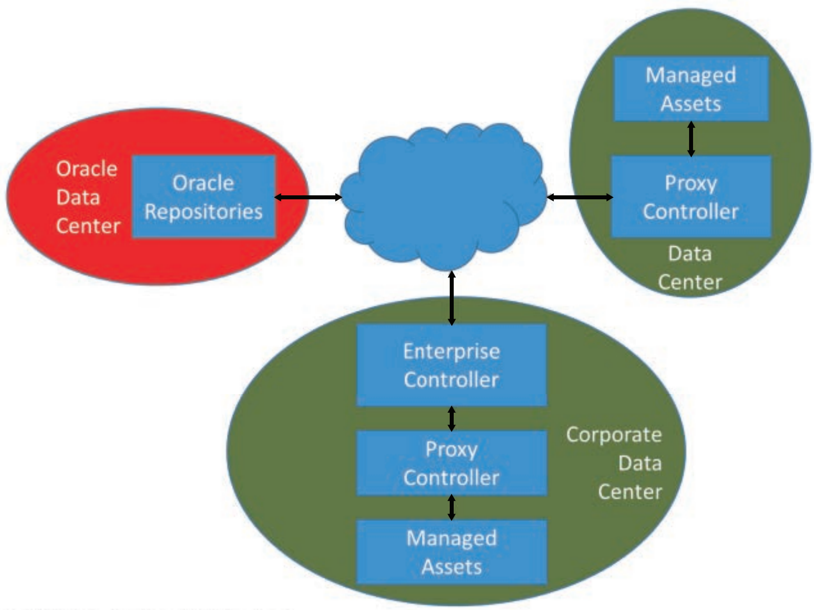


Figure 7.1 Ops Center Architecture

7.1.2 Virtualization Controllers

The Ops Center administrator can choose from two types of virtual environments. One type uses Solaris Zones; this type is simply called a global zone. The other type is a control domain, whose name refers to the use of OVM Server for SPARC. All systems that can be managed by Ops Center can be the global zone type. On modern SPARC systems, you can choose either a control domain or a global zone.

After you make that choice, Ops Center deploys the appropriate type of agent software in the management space, either the computer's control domain or global zone. This agent is called the virtualization controller (VC). Once its installation is complete, you can create the appropriate type of VEs on that server: logical domains for a control domain, or Solaris Zones for a global zone.

7.1.3 Control Domains

Control domains (CDoms) manage Oracle VM Server for SPARC logical domains (LDoms) on a computer. When you use Ops Center to provision a CDom, you choose the operating system, CDom hardware configuration (RAM, cores, and I/O), and names of virtual services provided to other domains. The service names include those for virtual disk services, network services, and console services. You can also initialize advanced Solaris features at the network layer for improved redundancy and performance, such as link aggregation. Advanced configurations, such as SR-IOV, service domains, and root complex domains, are also supported.

Once the CDom is provisioned, the Ops Center user can begin building guests. The guests must boot from either a virtual or physical disk. Using virtual disks provides the greatest flexibility at an extremely minimal performance cost. Virtual disks can reside on a number of physical media available to the CDom:

- A local file system
- A local disk
- An iSCSI LUN
- A NAS file system
- A FibreChannel LUN

When creating LDom guests, the Ops Center user creates one or more logical domain profiles that define the make-up of the guest:

- Name, CPU, core, and memory allocation
- Full core or vCPU allocation
- Architecture of the CPUs
- Networks
- Storage (local, iSCSI, NAS, FC)

When this information is combined with an operating system provisioning profile, the user can both create and provision one or more LDom guests quickly and easily by supplying a small amount of information, such as an IP address.

Further, the user can create a deployment plan to create multiple LDom guests with a single flow through the Ops Center user interface. After the deployment plan has been created, it can be used very easily to quickly create a large number of VEs, each ready to run a workload. Each of these guests will include all of the configuration details of the library image that was deployed, ensuring similarity for applications.

7.1.4 Global Zones

Global zones can be used to host applications, Solaris Zones, or any combination of those. Within the context of Ops Center, for a logical domain to include zones, the "global zone" agent must be installed in the LDom.

The Ops Center user may create a Solaris Zone profile that defines how zones will be created. Configuration options include the following:

- Dedicated or shared memory and CPU resources
- Type of zone (native or branded)
- Source of installation (e.g., operating system archive or network-based package source)
- Storage configuration (FC, iSCSI, or local disk)
- IP/Network configuration (exclusive or shared)
- DNS/Naming Services
- Time zone
- Root and administration passwords

Again, the user can create a deployment plan, based on a zone profile, to create multiple similar zones.

7.1.5 Storage Libraries

Ops Center tracks which LUNs and file systems are allocated to which guests, and ensures that more than one guest does not access the same LUN simultaneously. This constraint applies to both environments created with Ops Center and existing environments that are discovered by, and integrated into, Ops Center.

Ops Center manages this storage by using an underlying storage concept called storage libraries. Storage libraries are shared storage that is used for VEs, either for boot or data storage. Three types of storage can be used for storage libraries:

- NAS
- A static library, using LUNs created ahead of time:
  - FibreChannel
  - iSCSI
- A dynamic library, using a ZFS storage appliance, creating LUNs as needed

7.1.6 Server Pools

Ops Center includes another feature for virtualization that greatly enhances the automation, mobility, and recoverability of both LDom and zone environments—namely, a server pool. A server pool is a collection of similar VEs. It can be a group of zones or LDom hosts (CDoms), but not both types. A server pool of Solaris Zones must include servers with the same CPU technology, either SPARC or x86.

For a control domain server pool, Ops Center manages the placement of LDoms into physical computers using its own rules, guided by configuration information that the user provides and the current load on those computers. Ops Center can also dynamically balance the load periodically, among the servers in the pool.

A global zone server pool is treated the same way: Ops Center runs the zones in the servers, or LDoms, according to its rules and configuration information.

A server pool consists of the following components:

- Similar VEs
- Shared storage libraries (FC, NAS, iSCSI)
- Shared networks—a very small NAS share used to store guest metadata

The metadata comprises all of the information and resources for the guest. It is used for both migration and recovery scenarios.

Server pools enable two main mobility and recoverability features to be used in conjunction with virtualization—migration and automatic recovery.

7.1.7 Migration

Guests can migrate between hosts within a server pool. Depending on the underlying virtualization technology, this migration will either be “live” or “cold.” In live migration, the guest VE is moved to the destination without any interruption of the VE’s operation. In contrast, cold migration requires stopping the guest and restarting it on another host in the pool. Ops Center provides a simple way to automate the safe migration of guests from the central browser interface. It performs preflight checks to ensure that a guest can migrate and that the migration will succeed prior to initiating the actual migration step.

7.1.8 Automatic Recovery

Automatic recovery resolves a software or hardware failure without any user interaction. In the event of a server failure, guests on that member of the pool are automatically restarted on remaining, healthy hosts in the pool. Each guest that is no longer running will be automatically restarted on a healthy host in the pool.

For example, in a pool of five servers, imagine that Server 1 suffers a hardware fault and stops responding. Ops Center will restart the guest(s) that had been running on Server 1 on the remaining servers in the pool. Ops Center uses internal algorithms to determine which hosts are healthy and have sufficient resources. It uses placement rules provided when the pool was constructed to select the host on which each guest is restarted.

7.1.9 Layered Virtualization

Ops Center supports and helps automate a very popular "layered" virtualization technology. In this technology, one layer of virtualization runs underneath another layer.

The pool administrator can create a CDom server pool, where multiple LDoms are part of the pool. You can then use Ops Center to create multiple zones in one or more LDoms (see Figure 7.2). If you use layered virtualization, instead of migrating or automatically recovering at the zone level, those operations are handled at the LDom layer.

When you live migrate an LDom that has zones, the zones are automatically migrated with the LDom, and do not experience any downtime. When an LDom is automatically recovered, the zones will also be recovered and restarted automatically.

7.1.10 Summary

Virtualization technologies enable efficient consolidation, but require efficient management tools. Data center staff can use Oracle Enterprise Manager Ops Center to easily manage hundreds or thousands of VEs in multiple data centers, leveraging its efficient architecture to provision, monitor, and manage those guest VEs.

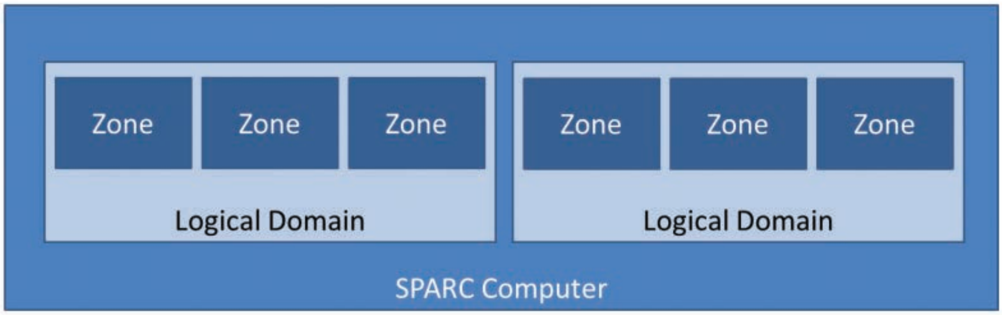


Figure 7.2 Layered Virtualization

7.2 OpenStack

A structured implementation of a private cloud would benefit from well-defined services, which are consumed by the virtual environments that self-service users deploy. One popular implementation of those services, along with the management tools necessary to deploy and use a private cloud, is OpenStack. The following subsections describe OpenStack briefly, and then discuss the integration of Oracle Solaris and OpenStack.

7.2.1 What Is OpenStack?

OpenStack is a community-based open-source project to form a comprehensive management layer to create and manage private clouds. This project was first undertaken as a joint effort of Rackspace and NASA in 2010, but is now driven by the OpenStack Foundation. Since 2010, OpenStack has been the fastest-growing open-source project on a worldwide basis, with thousands of commercial and individual contributors spread across the globe. The community launches two OpenStack releases per year.

OpenStack can be considered an operating system for cloud environments. It provides the foundation for Infrastructure as a Service (IaaS) clouds. Some new modules add features required in Platform as a Service (PaaS) clouds. OpenStack should not be viewed as layered software, however, but rather as an integrated infrastructure component. Thus, although the OpenStack community launches OpenStack releases, infrastructure vendors must integrate the open-source components into their own platforms to deliver the OpenStack functionality. Several operating system, network, and storage vendors offer OpenStack-enabled products.

OpenStack abstracts compute, network, and storage resources for the user, with those resources being exposed through a web portal with a single management pane. This integrated approach enables administrators to easily manage a variety of storage devices and hypervisors. The cloud services are based on a series of OpenStack modules, which communicate through a defined RESTful API between the various modules.

If a vendor plans to offer support for certain OpenStack services in its products, it must implement the functionality of those services and provide access to the functionality through the REST APIs. This can be done by delivering a service plugin, specialized for the product, that fills the gap between the REST API definition and the existing product feature.

7.2.2 The OpenStack General Architecture

Figure 7.3 depicts the general architecture of an OpenStack deployment. It consists of services provided by the OpenStack framework, and compute nodes that consume those services. This section describes those services.

Several OpenStack services are used to form an OpenStack-based private cloud. The services are interconnected via the REST APIs and depend on each other. But not all services are always needed to form a cloud, however, and not every vendor

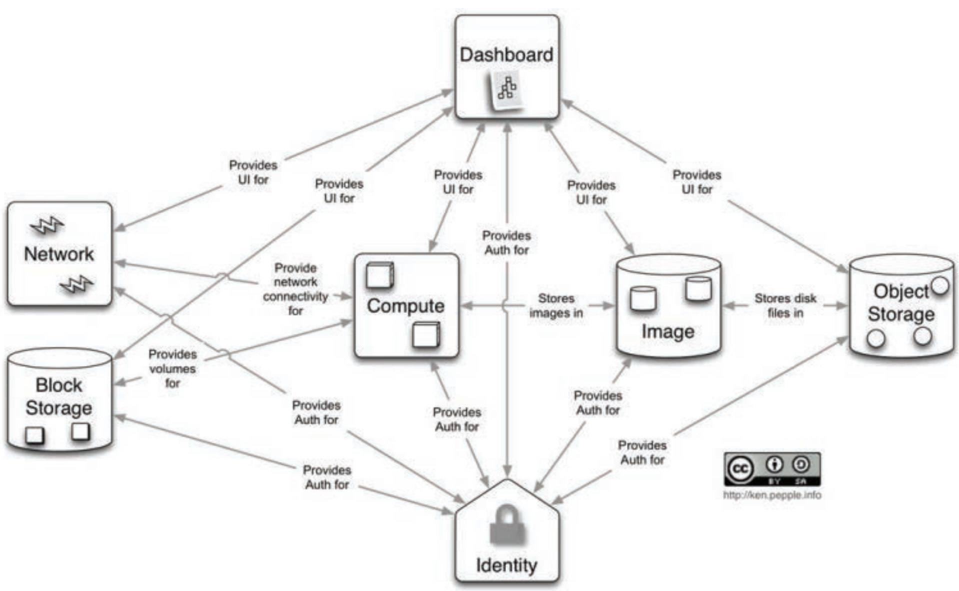


Figure 7.3 OpenStack Architecture

delivers all services. Some services have a special purpose and are configured only when appropriate; others are always needed when setting up a private cloud.

Because of the clearly defined REST APIs, services are extensible. The following list summarizes the core service modules.

- Cinder (block storage): Provides block storage for OpenStack compute instances and manages the creation, attaching, and detaching of block devices to OpenStack instances.
- Glance (images): Provides discovery, registration, and delivery services for disk and server images. The stored images can be used as templates for the deployment of VEs.
- Heat (orchestration): Enables the orchestration of complete application stacks, based on heat templates.
- Horizon (dashboard): Provides the dashboard management tool to access and provision cloud-based resources from a browser-based interface.
- Ironic (bare-metal provisioning): Used to provision bare-metal OpenStack guests, such as physical nodes.
- Keystone (authentication and authorization): Provides authentication and high-level authorization for the cloud and between cloud services. It consists of a central directory of users mapped to those cloud services they can access.
- Manila (shared file system): Allows the OpenStack instances to access shared file systems in the cloud.
- Neutron (network): Manages software-defined network services such as networks, routers, switches, and IP addresses to support multitenancy.
- Nova (compute): The primary service that provides the provisioning of virtual compute environments based on user requirement and available resources.
- Swift (object storage): A redundant and scalable storage system, with objects and files stored and managed on disk drives across multiple servers.
- Trove (database as a service): Allows users to quickly provision and manage multiple database instances without the burden of handling complex administrative tasks.

7.2.3 Oracle Solaris and OpenStack

Oracle Solaris 11 includes a full distribution of OpenStack as a standard, supported part of the platform. The first such release was Oracle Solaris 11.2, which integrated the Havana OpenStack release. The Juno release was integrated into

Oracle Solaris 11.2 Support Repository Update (SRU) 6. In Solaris 11.3 SRU 9, the integrated OpenStack software was updated to the Kilo release.

OpenStack services have been tightly integrated into the technology foundations of Oracle Solaris. The integration of OpenStack and Solaris leveraged many new Solaris features that had been designed specifically for cloud environments. Some of the Solaris features integrated into OpenStack include:

- Solaris Zones driver integration with Nova to deploy Oracle Solaris Zones and Solaris Kernel Zones
- Neutron driver integration with Oracle Solaris network virtualization, including Elastic Virtual Switch
- Cinder driver integration with the ZFS file system
- Unified Archives integration with Glance image management and Heat orchestration
- Bare-metal provisioning implementation using the Oracle Solaris Automated Installer (AI)

Figure 7.4 shows the OpenStack services implemented in Oracle Solaris and the related supporting Oracle Solaris features.

All services have been integrated into the Solaris Service Management Framework (SMF) to ensure service reliability, automatic service restart, and node

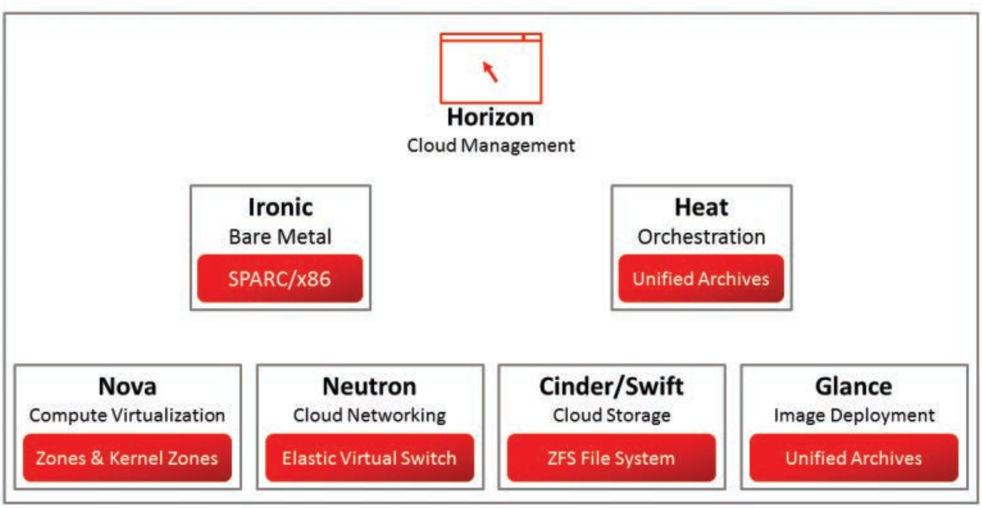


Figure 7.4 OpenStack Services in Oracle Solaris

dependency management. SMF properties enable additional configuration options. Oracle Solaris Role-Based Access Control (RBAC) ensures that the OpenStack services, represented by their corresponding SMF services, run with minimal privileges.

The OpenStack modules are delivered in separate Oracle Solaris packages, as shown in this example generated in Solaris 11.3:

```
# pkg list -af | grep openstack
cloud/openstack                 0.2015.2.2-0.175.3.9.0.2.0   i--
cloud/openstack/cinder          0.2015.2.2-0.175.3.9.0.2.0   i--
cloud/openstack/glance          0.2015.2.2-0.175.3.9.0.2.0   i--
cloud/openstack/heat            0.2015.2.2-0.175.3.9.0.2.0   i--
cloud/openstack/horizon         0.2015.2.2-0.175.3.9.0.2.0   i--
cloud/openstack/ironic          0.2015.2.1-0.175.3.9.0.2.0   i--
cloud/openstack/keystone        0.2015.2.2-0.175.3.9.0.2.0   i--
cloud/openstack/neutron         0.2015.2.2-0.175.3.9.0.2.0   i--
cloud/openstack/nova            0.2015.2.2-0.175.3.9.0.2.0   i--
cloud/openstack/openstack-common 0.2015.2.2-0.175.3.9.0.2.0  i--
cloud/openstack/swift           2.3.2-0.175.3.9.0.2.0        i--
```

To easily install the whole OpenStack distribution on a system, the `cloud/openstack` group package may be installed. It automatically installs all of the dependent OpenStack modules and libraries, plus additional packages such as `rad`, `rabbitmq`, and `mysql`.

The integration of OpenStack with the Solaris Image Packaging System (IPS) greatly simplifies updates of OpenStack on a cloud node, through the use of full package dependency checking and rollback. This was accomplished through integration with ZFS boot environments. Through a single update mechanism, an administrator can easily apply the latest software fixes to a system, including the virtual environments.

7.2.4 Compute Virtualization with Solaris Zones and Solaris Kernel Zones

Oracle Solaris Zones and Oracle Solaris Kernel Zones are used for OpenStack compute functionality. They provide excellent environments for application workloads and are fast and easy to provision in a cloud environment.

The life cycle of Solaris Zones as compute instances in an OpenStack cloud is controlled by the Solaris Nova driver for Solaris Zones. The instances are deployed by using the Nova command-line interface or by using the Horizon dashboard. To launch an instance, the cloud user selects a flavor, a Glance image, and a Neutron network. Figures 7.5 and 7.6 show the flavors available with Oracle Solaris OpenStack and the launch screen for an OpenStack instance.

☐	Flavor Name	VCPUs	RAM	Root Disk	Ephemeral Disk	Swap Disk	ID	Public	Metadata	Actions
☐	Oracle Solaris non-global zone - tiny	1	1GB	20GB	0GB	0MB	6	Yes	Yes	Edit Flavor
☐	Oracle Solaris kernel zone - tiny	1	2GB	20GB	0GB	0MB	1	Yes	Yes	Edit Flavor
☐	Oracle Solaris non-global zone - small	4	2GB	40GB	0GB	0MB	7	Yes	Yes	Edit Flavor
☐	Oracle Solaris kernel zone - small	4	4GB	40GB	0GB	0MB	2	Yes	Yes	Edit Flavor
☐	Oracle Solaris non-global zone - medium	8	4GB	80GB	0GB	0MB	8	Yes	Yes	Edit Flavor
☐	Oracle Solaris kernel zone - medium	8	8GB	80GB	0GB	0MB	3	Yes	Yes	Edit Flavor
☐	Oracle Solaris non-global zone - large	16	8GB	160GB	0GB	0MB	9	Yes	Yes	Edit Flavor
☐	Oracle Solaris kernel zone - large	16	16GB	160GB	0GB	0MB	4	Yes	Yes	Edit Flavor
☐	Oracle Solaris non-global zone - xlarge	32	16GB	320GB	0GB	0MB	10	Yes	Yes	Edit Flavor
☐	Oracle Solaris kernel zone - xlarge	32	32GB	320GB	0GB	0MB	5	Yes	Yes	Edit Flavor

Figure 7.5 OpenStack Flavors

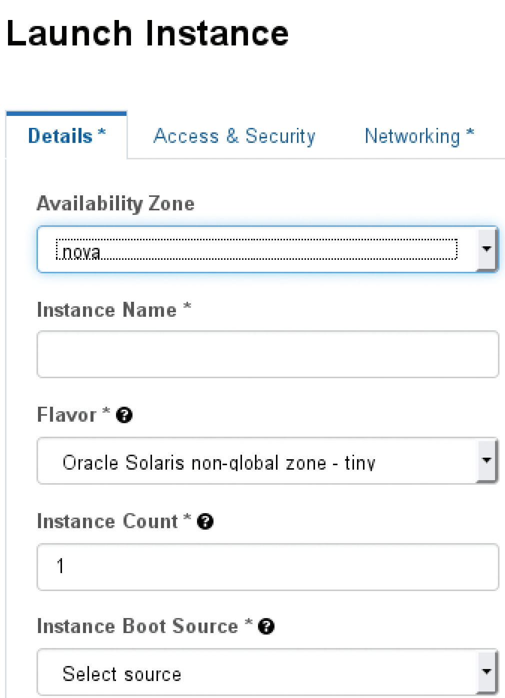


Figure 7.6 OpenStack Instance Launch Screen

Oracle Solaris options specify the creation of a Solaris native zone or a Solaris kernel zone. Those special properties are assigned as `extra_specs`, which are typically set through the command line. The property's keys comprise a set of zone properties that are typically configured with the `zonecfg` command and that are supported in OpenStack.

The following keys are supported in both kernel zones and non-global zone flavors:

- `zonecfg:bootargs`
- `zonecfg:brand`
- `zonecfg:hostid`
- `zonecfg:cpu-arch`

The following keys are supported only in non-global zone flavors:

- `zonecfg:file-mac-profile`
- `zonecfg:fs-allowed`
- `zonecfg:limitpriv`

The list of current flavors can be displayed on the command line:

```
+----+-----------------------------------------+-----------------------------------+
| ID | Name                                    | extra_specs                       |
+----+-----------------------------------------+-----------------------------------+
| 1  | Oracle Solaris kernel zone - tiny       | {u'zonecfg:brand': u'solaris-kz'} |
| 10 | Oracle Solaris non-global zone - xlarge | {u'zonecfg:brand': u'solaris'}    |
| 2  | Oracle Solaris kernel zone - small      | {u'zonecfg:brand': u'solaris-kz'} |
| 3  | Oracle Solaris kernel zone - medium     | {u'zonecfg:brand': u'solaris-kz'} |
| 4  | Oracle Solaris kernel zone - large      | {u'zonecfg:brand': u'solaris-kz'} |
| 5  | Oracle Solaris kernel zone - xlarge     | {u'zonecfg:brand': u'solaris-kz'} |
| 6  | Oracle Solaris non-global zone - tiny   | {u'zonecfg:brand': u'solaris'}    |
| 7  | Oracle Solaris non-global zone - small  | {u'zonecfg:brand': u'solaris'}    |
| 8  | Oracle Solaris non-global zone - medium | {u'zonecfg:brand': u'solaris'}    |
| 9  | Oracle Solaris non-global zone - large  | {u'zonecfg:brand': u'solaris'}    |
```

The `sc_profile` key can be modified only from the command line. This key is used to specify a system configuration profile for the flavor—for example, to preassign DNS or other system configurations to each flavor. For example, the following command will set a specific system configuration file for a flavor in the previously given list (i.e., "Oracle Solaris kernel zone – large"):

```
$ nova flavor-key 4 set sc_profile=/system/volatile/profile/sc_profile.xml
```

Launching an instance initiates the following actions in an OpenStack environment:

- The Nova scheduler selects a compute node in the cloud, based on the selected flavor, that meets the hypervisor type, architecture, number of VCPUs, and RAM requirements.
- On the chosen compute node, the Solaris Nova implementation will send a request to Cinder to find suitable storage in the cloud that can be used for the new instance's root file system. It then triggers the creation of a volume in that storage. Additionally, Nova obtains networking information and a network port in the selected network for an instance, by communicating with the Neutron service.
- The Cinder volume service delegates the volume creation to the storage device, receives the related Storage Unified Resource Identifier (SURI), and communicates that SURI back to the selected compute node. Typically this volume will reside on a different system from the compute node and will be accessed by the instance using shared storage such as FibreChannel, iSCSI, or NFS.
- The Neutron service assigns a Neutron network port to the instance, based on the cloud networking configuration. All instances instantiated by the compute service use an exclusive IP stack instance. Each instance includes an anet resource with its `configure-allowed-address` property set to `false`, and its `evs` and `vport` properties set to UUIDs supplied by Neutron that represent a particular virtualized switch segment and port.
- After the Solaris Zone and OpenStack resources have been configured, the zone is installed and booted, based on the assigned Glance image. This uses Solaris Unified Archives.

The following example shows a Solaris Zones configuration file, created by OpenStack for an iSCSI Cinder volume as boot volume:

```
compute-node # zonecfg -z instance-00000008 info
zonename: instance-00000008
brand: solaris
tenant: 740885068ed745c492e55c9e1c688472
anet:
        linkname: net0
        configure-allowed-address: false
        evs: a6365a98-7be1-42ec-88af-b84fa151b5a0
        vport: 8292e26a-5063-4bbb-87aa-7f3d51ff75c0
rootzpool:
        storage: iscsi://st01-sn:3260/target.iqn.1986-03.com.sun:02:...
capped-cpu:
        [ncpus: 1.00]
capped-memory:
        [swap: 1G]
rctl:
        name: zone.cpu-cap
        value: (priv=privileged,limit=100,action=deny)
rctl:
        name: zone.max-swap
        value: (priv=privileged,limit=1073741824,action=deny)
```

7.2.5 Cloud Networking with Elastic Virtual Switch

OpenStack networking creates virtual networks that interconnect VEs instantiated by the OpenStack compute node (Nova). It also connects these VEs to network services in the cloud, such as DHCP and routing. Neutron provides APIs to create and use multiple networks and to assign multiple VEs to networks, which are themselves assigned to different tenants. Each network tenant is represented in the network layer via an isolated Layer 2 network segment—comparable to VLANs in physical networks. Figure 7.7 shows the relationships among these components.

Subnets are properties that are assigned much like blocks of IPv4 or IPv6 addresses—that is, `default-router` or `nameserver`. Neutron creates ports in these subnets and assigns them together with several properties to virtual machines. The L3-router functionality of Neutron interconnects tenant networks to external networks and enables VEs to access the Internet through source NAT. Floating IP addresses create a static one-to-one mapping from a public IP address on the external network to a private IP address in the cloud, assigned to one VE.

Oracle Solaris Zones and Oracle Solaris Kernel Zones, as OpenStack instances, use the Solaris VNIC technology to connect to the tenant networks. All VNICs are bound with virtual network switches to physical network interfaces. If multiple

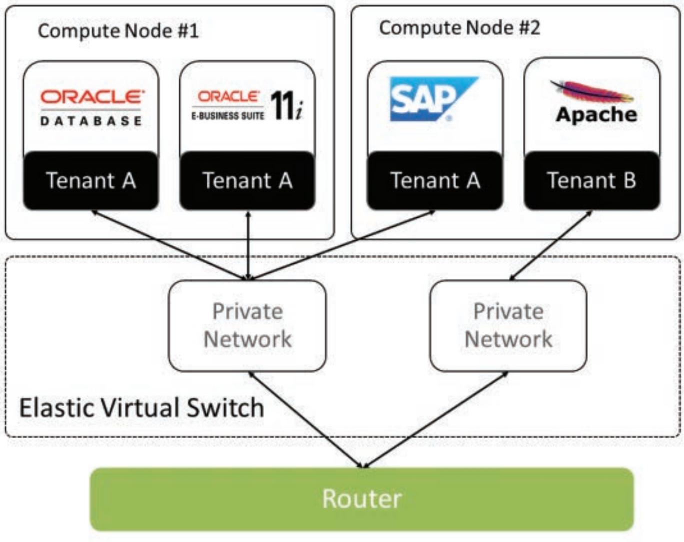


Figure 7.7 OpenStack Virtual Networking

tenants use one physical interface, then multiple virtual switches are created above that physical interface.

If multiple compute nodes have been deployed in one cloud and multiple tenants are used, virtual switches from the same tenant are spread over multiple compute nodes, as shown in Figure 7.8.

A technology is needed to control these distributed switches as one switch. The virtual networks can be created by, for example, VXLAN or VLAN. In the case of Oracle Solaris, the Solaris Elastic Virtual Switch (EVS) feature is used

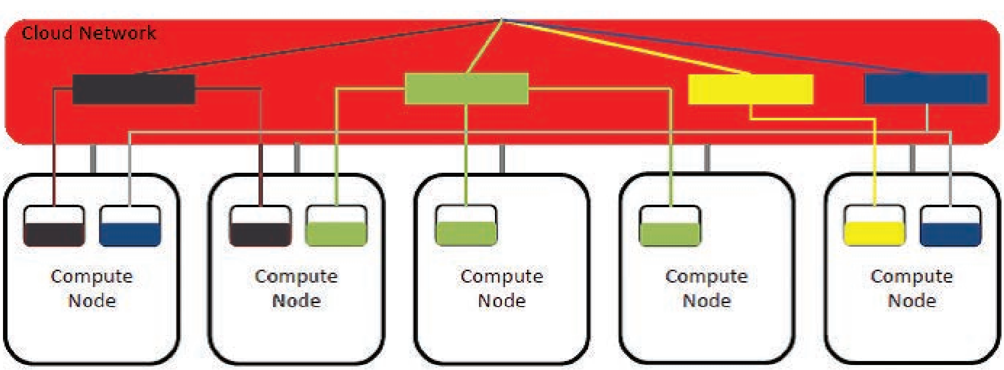


Figure 7.8 Virtual Switches

to control the distributed virtual switches. The back-end to OpenStack uses a Neutron plugin.

Finally, EVS is controlled by a Neutron plugin so that it offers an API to the cloud. In each compute node, the virtual switches are controlled by an EVS plugin to form a distributed switch for multiple tenants.

7.2.6 Cloud Storage with ZFS and COMSTAR

The OpenStack Cinder service provides central management for block storage volumes as boot storage and for application data. To create a volume, the Cinder scheduler selects a storage back-end, based on storage size and storage type requirements, and the Cinder volume service controls the volume creation. The Cinder API then sends the necessary access information back to the cloud.

Different types of storage can be used to provide storage to the cloud, such as FibreChannel, iSCSI, NFS, or the local disks of the compute nodes. The type used depends on the storage requirements. These requirements include characteristics such as capacity, throughput, latency and availability, and requirements for local storage or shared storage. Shared storage is required if migration of OpenStack instances between compute nodes is needed. Local storage may often be sufficient for short-term, ephemeral data. The cloud user is not aware of the storage technology that has been chosen, because the Cinder volume service represents the storage simply as a type of storage, not as a specific storage product model.

The Cinder volume service is configured to use an OpenStack storage plugin, which knows the specifics of a storage device. Example characteristics include the method to create a Cinder volume, and a method to access the data.

Multiple Cinder storage plugins are available for Oracle Solaris, which are based on ZFS to provide volumes to the OpenStack instances:

- The `ZFSVolumeDriver` supports the creation of local volumes for use by Nova on the same node as the Cinder volume service. This method is typically applied when using the local disks in compute nodes.
- The `ZFSISCSIDriver` and the `ZFSFCDriver` support the creation and export of iSCSI and FC targets, respectively, for use by remote Nova compute nodes. COMSTAR allows any Oracle Solaris host to become a storage server, serving block storage via iSCSI or FC.
- The `ZFSSAISCSIDriver` supports the creation and export of iSCSI targets from a remote Oracle ZFS Storage Appliance for use by remote Nova compute nodes.

In addition, other storage plugins can be configured in the Cinder volume service, if the storage vendor has provided the appropriate Cinder storage plugin. For example, the `OracleFSFibreChannelDriver` enables Oracle FS1 storage to be used in OpenStack clouds to provide FibreChannel volumes.

7.2.7 Sample Deployment Options

The functional enablement of Oracle Solaris for OpenStack is based on two main precepts. The first aspect is the availability and support of the OpenStack API with various software libraries and plugins in Oracle Solaris. The second aspect is the creation and integration of OpenStack plugins to enable specific Oracle Solaris functions in OpenStack. As discussed earlier, those plugins have been developed and provided for Cinder, Neutron, and Nova, as well as for Ironic.

Deploying an OpenStack-based private cloud with OpenStack for Oracle Solaris is similar to the setup of other OpenStack-based platforms.

- The design and setup of the hardware platform (server systems, network and storage) for the cloud are very important. Careful design pays off during the configuration and production phases for the cloud.
- Oracle Solaris must be installed on the server systems. The installation of Oracle Solaris OpenStack packages can occur with installation of Solaris—a process that can be automated with the Solaris Automated Installer.
- After choosing between the storage options, the storage node is installed and integrated into the cloud.
- The various OpenStack modules must be configured with their configuration files, yielding a full functional IaaS private cloud with OpenStack. The OpenStack configuration files are located in the `/etc/[cinder`, `neutron`, `nova`, ..] directories. The final step is the activation of the related SMF services with their dependencies.

The design of the hardware platform is also very important. Besides OpenStack, a general cloud architecture to be managed by OpenStack includes these required parts:

- One or multiple compute nodes for the workload.
- A cloud network to host the logical network internal to the cloud. Those networks link together network ports of the instances, which together form

one network broadcast domain. This internal logical network is typically composed with VxLAN or tagged VLAN technology.

- Storage resources to boot the OpenStack instances and keep application data persistent.
- A storage network, if shared storage is used, to connect the shared storage with the compute nodes.
- An internal control network, used by the OpenStack API's internal messages and to drive the compute, network, and storage parts of the cloud; this network can also be used to manage, install, and monitor all cloud nodes.
- A cloud control part, which runs the various OpenStack control services for the OpenStack cloud like the Cinder and Nova scheduler, the Cinder volume service, the MySQL management database, or the RabbitMQ messaging service.

Figure 7.9 shows a general OpenStack cloud, based on a multinode architecture with multiple compute nodes, shared storage, isolated networks and controlled cloud access through a centralized network node.

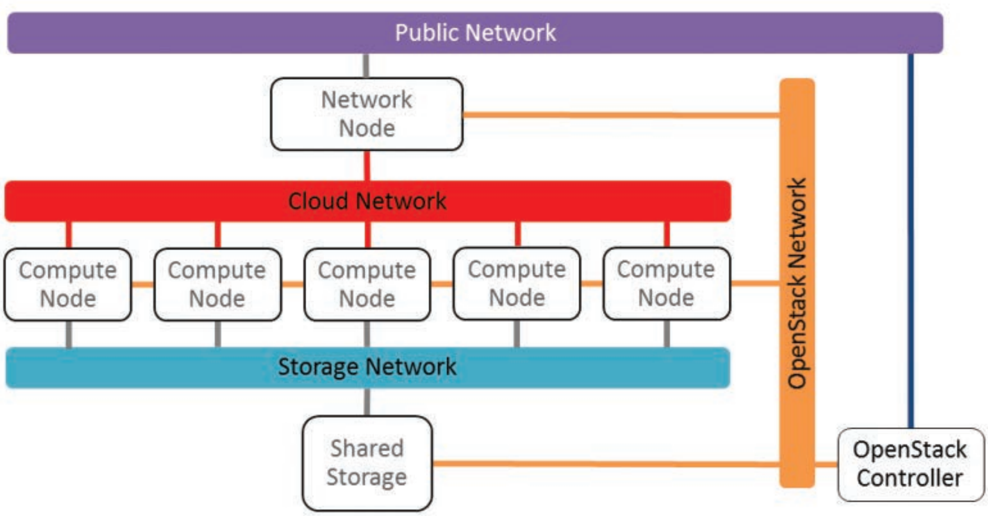


Figure 7.9 Single Public Network Connection

7.2.8 Single-System Prototype Environment

You can demonstrate an OpenStack environment in a single system. In this case, a single network is used, or multiple networks are created using etherstubs, to form the internal network of the cloud. "Compute nodes" can then be instantiated as kernel zones. However, if you use kernel zones as compute nodes, then OpenStack instances can be only non-global zones. This choice does not permit application of several features, including Nova migration. This single-node setup can be implemented very easily with Oracle Solaris, using a Unified Archive of a comprehensive OpenStack installation.

Such single-system setups are typically implemented so that users can become familiar with OpenStack or to create very small prototypes. Almost all production deployments will use multiple computers to achieve the availability goals of a cloud.

There is one exception to this guideline: A SPARC system running Oracle Solaris (e.g., SPARC T7-4) can be configured as a multinode environment, using multiple logical domains, connected with internal virtual networks. The result is still a single physical system, which includes multiple isolated Solaris instances, but is represented like a multinode cloud.

7.2.9 Simple Multinode Environment

Creating a multinode OpenStack cloud increases the choices available in all parts of the general cloud architecture. The architect makes the decision between one unified network or separate networks when choosing the design for the cloud network, the internal network, and the storage network. Alternatively, those networks might not be single networks, but rather networks with redundancy features such as IPMP, DLMP, LACP, or MPXIO. All of these technologies are part of Oracle Solaris and can be selected to create the network architecture of the cloud.

Another important decision to be made is how to connect the cloud to the public or corporate network. The general architecture described earlier shows a controlled cloud access through a centralized network node. While this setup enforces centralized access to the cloud via a network node, it can also lead to complicated availability or throughput limitations. An alternative setup is a flat cloud, shown in Figure 7.10, in which the compute nodes are directly connected to the public network, so that no single access point limits throughput or availability. It is the responsibility of the cloud architect to decide which option is the most appropriate choice.

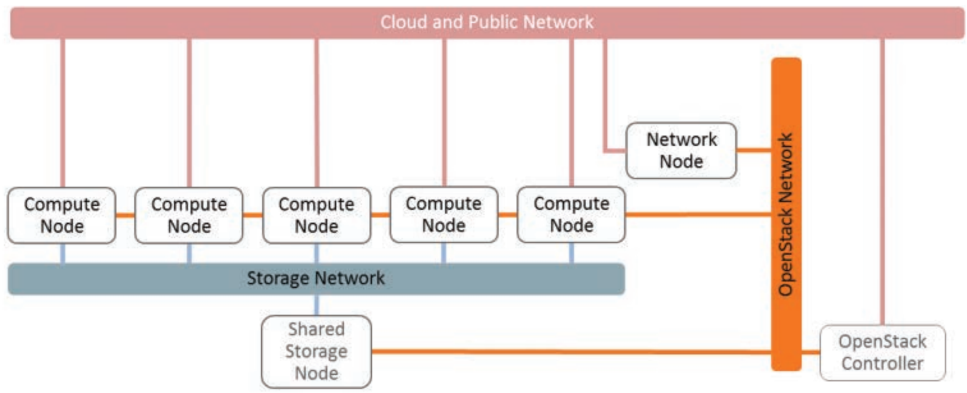


Figure 7.10 Multiple Public Network Connections

For the compute nodes, the decision can be made between SPARC nodes (SPARC T5, T7, S7, M7, or M10 servers), x86_64 nodes, or a mixed-node cloud that combines both architectures. Oracle Solaris OpenStack will handle both processor architectures in one cloud. Typically, compute nodes with 1 or 2 sockets with medium memory capacity (512 GB) are chosen. More generally, by using SPARC systems, compute nodes ranging from very small to very large in size can be combined in one cloud without any special configuration efforts.

The cloud storage is typically shared storage. In a shared storage architecture, disks storing the running instances are located outside the compute nodes. Cloud instances can then be easily recovered with migration or evacuation, in case of compute node downtime. Using shared storage is operationally simple because having separate compute hosts and storage makes the compute nodes "stateless." Thus, if there are no instances running on a compute node, that node can be taken offline and its contents erased completely without affecting the remaining parts of the cloud. This type of storage can be scaled to any amount of storage. Storage decisions can be made based on performance, cost, and availability. Among the choices are an Oracle ZFS storage appliance, shared storage through a Solaris node as iSCSI or FC target server, or shared storage through a FibreChannel SAN storage system.

To use local storage, each compute node's internal disks store all data of the instances that the node hosts. Direct access to disks is very cost-effective, because there is no need to maintain a separate storage network. The disk performance on each compute node is directly related to the number and performance of existing local disks. The chassis size of the compute node will limit the number of spindles able to be used in a compute node. However, if a compute node fails, the instances on it cannot be recovered. Also, there is no method to migrate instances. This omission can be a major issue for cloud services that create persistent data. Other cloud services, however, perform processing services without storing any local data, in which case no local persistent data is created.

The cloud control plane, implemented as an OpenStack controller, can consist of one or more systems. With Oracle Solaris, typically the OpenStack controller is created in kernel zones for modular setups. Scalability on the controller site can then be achieved just by adding another kernel zone. The OpenStack control services can all be combined in one kernel zone. For scalability and reliability reasons, the services can be grouped into separate kernel zones, providing the following services:

- RabbitMQ
- MySQL management database
- EVS Controller
- Network Node
- The remaining OpenStack Services

7.2.10 OpenStack Summary

Running OpenStack on Oracle Solaris provides many advantages. A complete OpenStack distribution is part of the Oracle Solaris Repository and, therefore, is available for Oracle Solaris without any additional cost. The tight integration of the comprehensive virtualization features for compute and networking—Solaris Zones, virtual NICs and switches, and the Elastic Virtual Switch—in Oracle Solaris provide significant value not found in other OpenStack implementations. The integration of OpenStack with Oracle Solaris leverages the Image Packaging System, ZFS boot environments, and the Service Management Facility. As a consequence, an administrator can quickly start an update of the cloud environment, and can quickly update each service and node in a single operation.

7.3 Summary

At least in IT, perfection cannot be achieved: Every solution inevitably yields new problems. In keeping with this pattern, virtualization was a solution, but it created a problem—the unbridled proliferation of virtual environments, a herd of invisible horses that must somehow be corralled.

In some cases, software intended to manage physical computers was extended with the ability to manage both physical computers and VEs. OEM Ops Center is an excellent example. New software such as OpenStack was also designed from the ground up to manage VEs and their surrounding infrastructure. Understanding these tools is a prerequisite to successful deployment of virtualization.

Index