Building Highly Available Applications in a Region with One Availability Domain

Disclaimer

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle software license and service agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced, or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement, nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

This document is for informational purposes only and is intended solely to assist you in planning for the implementation and upgrade of the product features described. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described in this document remains at the sole discretion of Oracle. Due to the nature of the product architecture, it may not be possible to safely include all features described in this document without risking significant destabilization of the code.

Revision History

The following revisions have been made to this document since its initial publication.

DATE	REVISION
May 2024	• Replaced link to bastion technical paper with link to the OCI Bastion service documentation • Updated the list of OCI regions
May 2023	• Reviewed and verified information • Updated the list of OCI regions
May 2022	Corrected statements about power sources in fault domains
November 2021	• Added notes about Terraform version • Updated the template
September 2019	Initial publication

ORACLE

Table of Contents

ORACLE

Introduction

Oracle Cloud Infrastructure (OCI) is rapidly deploying regions across the globe. Regions contain availability domains, and each availability domain contains fault domains to provide redundancy and isolation within the region. Fault domains provide the fault tolerance for network, server, and maintenance issues that can occur within a region. They provide the redundancy that enables applications to be highly available within any one availability domain.

Deploying a highly available application in OCI is a fundamental part of building an enterprise-ready environment application system in all the regions that are important to your business. This technical paper discusses the steps for building high availability into the applications being deployed across multiple fault domains in an availability domain. This paper is intended for system architects who want to understand how to build automated deployments of high-availability applications within OCI.

Note: The Terraform deployment code provided in this paper was written using Terraform version 0.10.

Solution Overview

OCI is composed of regions, availability domains, and fault domains. This design provides the fault tolerance and redundancy needed to ensure a robust computing environment. A region contains specific resources, such as a virtual cloud network (VCN). This basic building block provides the foundation for OCI to provide more resources that are wrapped within availability domains to enable a modular approach for expansion. The redundancy built into a region begins with a single availability domain that contains multiple fault domains.

Regions

A region covers a geographical area, such as a city or country. The following table lists the regions in the OC1 commercial realm and the number of availability domains in each region, as of the latest publication of this paper. The number of regions is growing rapidly; see the OCI documentation for the up-to-date list.

REGION NAME	REGION IDENTIFIER	REGION LOCATION	REGION KEY	AVAILABILITY DOMAINS
Australia East (Sydney)	ap-sydney-1	Sydney, Australia	SYD	1
Australia Southeast (Melbourne)	ap-melbourne-1	Melbourne, Australia	MEL	1
Brazil East (Sao Paulo)	sa-saopaulo-1	Sao Paulo, Brazil	GRU	1
Brazil Southeast (Vinhedo)	sa-vinhedo-1	Vinhedo, Brazil	VCP	1
Canada Southeast (Montreal)	ca-montreal-1	Montreal, Canada	YUL	1
Canada Southeast (Toronto)	ca-toronto-1	Toronto, Canada	YYZ	1
Chile Central (Santiago)	sa-santiago-1	Santiago, Chile	SCL	1
Chile West (Valparaiso)	sa-valparaiso-1	Valparaiso, Chile	VAP	1
Colombia Central (Bogota)	sa-bogota-1	Bogota, Colombia	BOG	1
France Central (Paris)	eu-paris-1	Paris, France	CDG	1
France South (Marseille)	eu-marseille-1	Marseille, France	MRS	1

ORACLE

REGION NAME	REGION IDENTIFIER	REGION LOCATION	REGION KEY	AVAILABILITY DOMAINS
Germany Central (Frankfurt)	eu-frankfurt-1	Frankfurt, Germany	FRA	3
India South (Hyderabad)	ap-hyderabad-1	Hyderabad, India	HYD	1
India West (Mumbai)	ap-mumbai-1	Mumbai, India	BOM	1
Israel Central (Jerusalem)	il-jerusalem-1	Jerusalem, Israel	MTZ	1
Italy Northwest (Milan)	eu-milan-1	Milan, Italy	LIN	1
Japan Central (Osaka)	ap-osaka-1	Osaka, Japan	KIX	1
Japan East (Tokyo)	ap-tokyo-1	Tokyo, Japan	NRT	1
Mexico Central (Queretaro)	mx-queretaro-1	Queretaro, Mexico	QRO	1
Mexico Northeast (Monterrey)	mx-monterrey-1	Monterrey, Mexico	MTY	1
Netherlands Northwest (Amsterdam)	eu-amsterdam-1	Amsterdam, Netherlands	AMS	1
Saudi Arabia West (Jeddah)	me-jeddah-1	Jeddah, Saudi Arabia	JED	1
Singapore (Singapore)	ap-singapore-1	Singapore, Singapore	SIN	1
South Africa Central (Johannesburg)	af-johannesburg-1	Johannesburg, South Africa	JNB	1
South Korea Central (Seoul)	ap-seoul-1	Seoul, South Korea	ICN	1
South Korea North (Chuncheon)	ap-chuncheon-1	Chuncheon, South Korea	YNY	1
Spain Central (Madrid)	eu-madrid-1	Madrid, Spain	MAD	1
Sweden Central (Stockholm)	eu-stockholm-1	Stockholm, Sweden	ARN	1
Switzerland North (Zurich)	eu-zurich-1	Zurich, Switzerland	ZRH	1
UAE Central (Abu Dhabi)	me-abudhabi-1	Abu Dhabi, UAE	AUH	1
UAE East (Dubai)	me-dubai-1	Dubai, UAE	DXB	1
UK South (London)	uk-london-1	London, United Kingdom	LHR	3
UK West (Newport)	uk-cardiff-1	Newport, United Kingdom	CWL	1
US East (Ashburn)	us-ashburn-1	Ashburn, VA	IAD	3
US Midwest (Chicago)	us-chicago-1	Chicago, IL	ORD	3
US West (Phoenix)	us-phoenix-1	Phoenix, AZ	PHX	3
US West (San Jose)	us-sanjose-1	San Jose, CA	SJC	1

ORACLE

Availability Domains

Think of an availability domain as a specific grouping of resources within a region that has redundant power, network, and compute. An availability domain is considered a separate data center, with all the separate resources of an independent data center within a specific geographical region. The data center has utility-provided power and backup generation to ensure that power is always available.

Fault Domains

Each availability domain contains three fault domains to provide redundancy within the availability domain. Fault domains are groups of hardware and infrastructure services that don't share network or maintenance events. Fault domains allow instances to be distributed to protect against hardware failures. Using the data center example, fault domains have separated hardware racks. If an application is spread across fault domains, and a rack or hardware failure occurs in one of the fault domains, the application can fail over quickly to hardware in a different fault domain.

Key Considerations for Deploying in a Region with One Availability Domain

When you deploy to a region that has only one availability domain, it's important to understand the difference between the resources that are regional (in the availability domain) and the resources in the fault domain.

- Regional resources include VCNs, DHCP options, internet gateways, load balancers, service gateways, route tables, and security lists.

- Availability domain–specific resources include instances, subnets, and volumes.

- Fault domains contain the distinct physical hardware, such as bare metal hardware.

For a complete list of what services are provided among regions, availability domains, and fault domains, see the documentation.

When planning the architecture for an application, ensure that the network is segmented to allow for expansion and the necessary security to protect the compute assets. After building the network plan, identify the assets and how they're associated within the region, availability domain, and fault domains to ensure that compute nodes are balanced across the fault domains. Plan for management access, and place those items in the proper subnets. Security lists are crucial to ensure that proper access and the right user profiles have access to the data and applications.

OCI uses Terraform to build and maintain environments. It builds an initial configuration and maintains that configuration to help reduce deployment drift. Terraform is excellent for building and managing the network, from the VCN to the subnets.

Terraform lets you programmatically manage, version, and persist IT Infrastructure as infrastructure-as-code. By using Terraform's declarative syntax, the infrastructure can be maintained and stored as code in a code repository, which allows for versioning and the distribution of the infrastructure as code. To read about supported features and best practices, see the Terraform provider for OCI documentation.

ORACLE

Solution Architecture

To illustrate the basics of building a highly available application in an OCI region with one availability domain, consider the architecture of a simple web service application with Microsoft Internet Information Services (IIS). This application uses an OCI load balancer and two Windows servers, each with an IIS server that is connected to a File Storage file share.
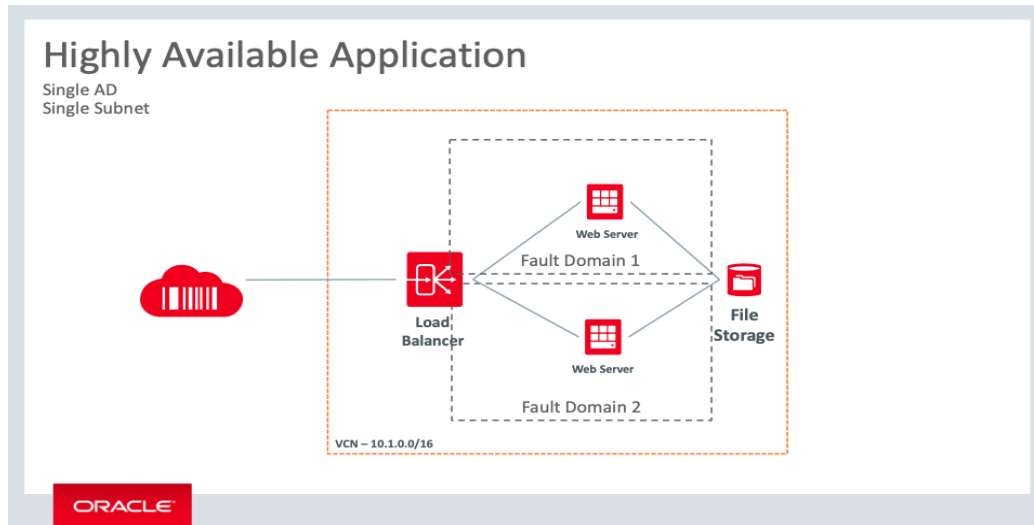

Figure 1: Highly Available Web Service Application

This basic structure shows the client interaction with the load balancer, which round-robins clients between the web servers that rely on a backend file system. The compute resources are contained in a single, private subnet that protects the web servers, and each server is in a separate fault domain to ensure high availability. The following table explains the association of the resources.

RESOURCE NAME	PRODUCT CATEGORY	SCOPE	DOCUMENTATION
Windows Server	Compute	Fault domain	All Image Families
File system	File Storage	Region	Overview of File Storage
Load balancer	Networking	Region	Overview of Load Balancing
VCN	Networking	Region	VCNs and Subnets
Subnets	Networking	Region	VCNs and Subnets

Note: Some other solution-management components aren't discussed in this paper, including bastion servers for management, demilitarized zone (DMZ) setup, and Active Directory setup. For more information, see the links in the "Resources" section at the end of this paper.

The architecture can be divided into three key components: networking, compute, and file storage. This section discusses the relationships among these components and how they relate to the OCI region and availability domain structure. To create an environment that's secure and ready to scale, ensure that the network is set up for growth and flexibility while maintaining security to protect the assets within the environment. Terraform is used to build the environment and help to maintain an initial state for the high-availability solution.

ORACLE

Networking

The network is the foundation for the solution. The network consists of VCNs, load balancers, subnets, internet gateways, security lists, and route tables. The VCN is the main network in the region and is a part of the compartment. The VCN must be large enough to contain all the necessary components. This solution uses a 10.1.0.0/16 network space, which allows for growth; the /16 segmentation allows for 65,534 IP addresses within the VCN. You can use several IP subnet calculators to plan the size of the required network. Dividing the subnets further to build a region-spanning subnet with a range on 10.1.20.0/24 gives a total of 254 usable devices within that subnet. The compute components are located within this regional subnet.

The subnets need security lists defined to ensure proper access for applications and users, and routing tables need the correct routes to connect to all the necessary resources. These attributes are discussed more in the "Deployment Code" section, which describes the use of Terraform to build the solution configuration.

The load balancer is used to round-robin connections to the IIS servers to enable client access to the web service. The load balancer has endpoint connections that are part of the VCN component and aren't inside the subnets. This abstraction provides a layer of segmentation to protect the compute nodes from direct attack. The load balancer gives a single entry to multiple servers within the VCN. It helps to reduce management through policies and uses application health checks to direct network traffic to the appropriate web servers. The load balancer enables the creation of public or private access points for this solution. It uses a public internet address and multiple listeners to balance Layer 4 and Layer 7 traffic. This single availability domain solution requires a private IP address from the host subnet and a floating IP address.

Compute

The compute resources for this solution are two Windows Server 2016 Standard servers running the Oracle-provided image for the virtual machines (VMs). For more information about OCI compute shapes, see the documentation. Pick the right shape for the workload that fits your compute needs based on price and performance. Then configure IIS for the Windows server to serve as the primary application for this solution.

File Storage

The File Storage service is a robust file storage solution that's ideal for general-purpose file storage or high-performance workloads. Clients can connect to File Storage within the VCN, with FastConnect, or with Internet Protocol Security (IPSec) over a virtual private network (VPN). The architecture for this solution provides a backend for the IIS web service by using Network File System version 3.0 (NFSv3) and supports the network lock manager (NLM) protocols. Integrated with the NFS client for Windows, File Storage gives performant file access across multiple IIS servers, providing a scalable web service.

Deployment Code

Note: The Terraform deployment code provided in this paper was written using Terraform version 0.10.

The code for this deployment of a simple high-availability application is separated into the following core files:

- `variables.tf` (Appendix A: Variables.tf)
- `main.tf` (Appendix B: Main.tf)
- `bootvars.ps1` (Appendix C: Bootvars.ps1)

ORACLE

Each file is provided in its entirety in the appendices of this paper. This section explains the operations, the code, and how they work together.

The first component deployed is the network, creating the VCN and subnets. The Terraform provider then creates the file system, starts the load balancer, and creates the VMs. Terraform creates the objects in the correct order; the variables and code in the files don't need to be in a specific order. The rest of this section discusses the secrets and providers, network and hosts, file system, and load balancer. The examples in this section aren't designed to be copied.

Secrets and Providers

The `variables.tf` file specifies the items that are needed throughout the deployment of the environment. Gather key items like the region code, compartment OCID, tenancy OCID, and user OCID. You also need user fingerprints and SSH keys to ensure proper security and use of tenancy resources. For more information, see the documentation.

You can find the tenancy OCID, region code, compartment OCID, and user OCID in the Oracle Cloud Console.

- Find the tenancy OCID and region code in **Tenancy details** under **Administration**.

- Find the compartment OCID in **Compartments** under **Identity.**

- Find the user OCID in **Users** under **Identity**.

Ensure that these values are inserted into the correct sections of the variables file.

These variables help determine the input needs of the provider API call from Terraform. This call sets the context for the rest of the actions of the Terraform provider.

```
provider "oci" {
version = ">= 3.0"
tenancy_ocid = "${var.tenancy_ocid}"
user_ocid = "${var.user_ocid}"
fingerprint = "${var.fingerprint}"
private_key_path = "${var.private_key_path}"
region = "${var.region}"
}
```

Network and VMs

After determining the required OCIDs and secrets, identify the network and host names that will be deployed in the region. We recommend that you have a descriptive and meaningful naming schema for hosts. It's helpful to indicate either the location or type of environment that's being deployed.

```
variable vcn_name {
default = "WebAppVCN01"
}
variable subnet_name {
default = "WebAppSubnet01"
}
```

ORACLE

The network provider covers the VCN, subnets, ingress and egress rules, security lists, and route table.

```
resource "oci_core_virtual_network" "vcn1" {
cidr_block = "10.1.0.0/16"
compartment_id = "${var.compartment_id}"
display_name = "${var.vcn_name}"
dns_label = "${var.vcn_name}"
}


resource "oci_core_subnet" "subnet1" {
availability_domain = "${var.availability_domain}"
cidr_block = "10.1.20.0/24"
display_name = "${var.subnet_name}"
dns_label = "${var.subnet_name}"
security_list_ids = ["${oci_core_security_list.securitylist1.id}"]
compartment_id = "${var.compartment_id}"
vcn_id = "${oci_core_virtual_network.vcn1.id}"
route_table_id = "${oci_core_route_table.routetable1.id}"
dhcp_options_id = "${oci_core_virtual_network.vcn1.default_dhcp_options_id}"

resource "oci_core_internet_gateway" "internetgateway1" {
compartment_id = "${var.compartment_id}"
display_name = "internetgateway1"
vcn_id = "${oci_core_virtual_network.vcn1.id}"
}


resource "oci_core_route_table" "routetable1" {
compartment_id = "${var.compartment_id}"
vcn_id = "${oci_core_virtual_network.vcn1.id}"
display_name = "routetable1"

route_rules {
destination = "0.0.0.0/0"
destination_type = "CIDR_BLOCK"
network_entity_id = "${oci_core_internet_gateway.internetgateway1.id}"
}
}

resource "oci_core_security_list" "securitylist1" {
display_name = "public"
compartment_id = "${oci_core_virtual_network.vcn1.compartment_id}"
vcn_id = "${oci_core_virtual_network.vcn1.id}"

egress_security_rules = [{

protocol = "all"
destination = "0.0.0.0/0"
}]

ingress_security_rules = [
{
protocol = "6"
source = "0.0.0.0/0"
```

ORACLE

```
tcp_options {
"min" = 80
"max" = 80
}
},

]


}
```

The VM requires a host name, the instance shape, and the instance image OCID. Verify valid shapes and image OCIDs in the [documentation](#). Using the map function lets you deploy the Oracle-provided image. You can replace this OCID with a custom image OCID.

```
variable hostname_1 {
default = "WebInstance1"
}

variable instance_shape {
default = "VM.Standard2.1"
}
```

Use the `map` variable when there's a possibility of running the Terraform in multiple regions. The instance image provided by OCI is tied to each separate region.

```
variable "instance_image_ocid" {
type = "map"

default = {
us-phoenix-1 = "ocid1.image.oc1.phx.aaaaaaaatsanuoggs6vzjwhtbmzficdr5364glhroki2o6wp74jva73ldcnq"
us-ashburn-1 = "ocid1.image.oc1.iad.aaaaaaaaq3l2t5p3i2ai6wkwqvi6fltnew4ctxih54kd5fukh4jehe45mnfq"
eu-frankfurt-1 = "ocid1.image.oc1.eu-frankfurt-1.aaaaaaaaz7zbcqrwkjvi7otldol6ael4qgb56mtvpu3favkwr5einc7zxycq"
uk-london-1 = "ocid1.image.oc1.uk-london-1.aaaaaaaa6jg4djqfofy2xa44ekcsbd6folvqsgoak4jjdrli2qbtwvtqmlva"
ca-toronto-1 = "ocid1.image.oc1.ca-toronto-1.aaaaaaaaadtsgda5axsza4nggqpxzc6lymtap3bxi5x7zjl3ohr32t4asnqa"
ap-seoul-1 = "ocid1.image.oc1.ap-seoul-1.aaaaaaaafcxeuvhwikas37miuycyqsrnoxmng44dld2iaxtuetxm3bak7byq"
ap-tokyo-1 = "ocid1.image.oc1.ap-tokyo-1.aaaaaaaaivx5l3xcvtnwyutsg3bo3sg4scw2yjchkoodxrjszak3mctxjq4q"
}
}
```

The availability domain information is shown as a header under the Service Limits page in the Console. Include the fault domain for each instance. The default uses the first fault domain, resulting in a potential loss of service to the application.

```
variable availability_domain {
default = "AaRH:CA-TORONTO-1-AD-1"
}

variable instance_fault_domain_1 {
default = "FAULT-DOMAIN-1"
}

variable instance_fault_domain_2 {
default = "FAULT-DOMAIN-2"
}
```

ORACLE

The VM instance Terraform provider code uses the information about the availability domain, fault domains, compartment, and subnet variables to place the VMs in the correct availability domain and fault domains. The user data PowerShell script finalizes the VMs after creation.

```
resource "oci_core_instance" "instance1" {
availability_domain = "${var.availability_domain}"
fault_domain = "${var.instance_fault_domain_1}"
compartment_id = "${var.compartment_id}"
display_name = "${var.hostname_1}"
shape = "${var.instance_shape}"
subnet_id = "${oci_core_subnet.subnet1.id}"
hostname_label = "${var.hostname_1}"

metadata {
user_data = "${base64encode(file("/Users/jsparker/Desktop/Code/HASingleADReg/lb/bootvars.ps1"))}"
}

source_details {
source_type = "image"
source_id = "${var.instance_image_ocid[var.region]}"
}
}

resource "oci_core_instance" "instance2" {
availability_domain = "${var.availability_domain}"
fault_domain = "${var.instance_fault_domain_2}"
compartment_id = "${var.compartment_id}"
display_name = "${var.hostname_2}"
shape = "${var.instance_shape}"
subnet_id = "${oci_core_subnet.subnet1.id}"
hostname_label = "${var.hostname_2}"

metadata {
user_data = "${base64encode(file("/Users/jsparker/Desktop/Code/HASingleADReg/lb/bootvars.ps1"))}"
}

source_details {
source_type = "image"
source_id = "${var.instance_image_ocid[var.region]}"
}
}
```

Appendix C: Bootvars.ps1 contains the user data PowerShell script. This script adds the web management tools, web server, and NFS client, and it imports the IIS and web server administration PowerShell modules. It starts the basic configuration of the IIS server.

```
Add-WindowsFeature Web-Mgmt-Tools, Web-Server
Install-WindowsFeature -Name NFS-Client
Import-Module IISAdministration
Import-Module WebAdministration

$IISFeatures = "<< Insert comma list of IIS features such as - Web-WebServer", "Web-Common-Http",...
etc. >>"
Install-WindowsFeature -Name $IISFeatures
```

ORACLE

These settings are basic. We recommend adjusting the web server's configuration to meet the application's needs.

File System and Load Balancer

The next step is building the File Storage services, which are availability domain resources, not fault domain resources. You manage the configuration of the file system in the variables section and set them according to the needs of the application server.

```
variable "mount_target_1_display_name" {
default = "NewFileSystem"
}
variable "file_system_1_display_name" {
default = "NewFileSystem1"
}
variable "export_set_name_1" {
default = "ExportedNFS"
}
variable "export_path_fs1_mt1" {
default = "/Newfilesystem1"
}
variable "export_read_write_access_source" {
default = "10.1.0.0/16"
}
variable "export_read_only_access_source" {
default = "0.0.0.0/0"
}
variable "max_byte" {
default = 23843202333
}

variable "max_files" {
default = 223442
}
```

For File Storage, the compartment, availability domain, and subnet are global variables. You don't need to separately define them.

```
resource "oci_file_storage_file_system" "my_fs_1" {
#Required
availability_domain = "${var.availability_domain}"
compartment_id = "${var.compartment_id}"

#Optional
display_name = "${var.file_system_1_display_name}"
}

resource "oci_file_storage_mount_target" "my_mount_target_1" {
#Required
availability_domain = "${var.availability_domain}"
compartment_id = "${var.compartment_id}"
subnet_id = "${oci_core_subnet.subnet1.id}"

#Optional
display_name = "${var.mount_target_1_display_name}"
}
```

ORACLE

```
resource "oci_file_storage_export_set" "my_export_set_1" {
# Required
mount_target_id = "${oci_file_storage_mount_target.my_mount_target_1.id}"

# Optional
display_name = "${var.export_set_name_1}"
# max_fs_stat_bytes = "${var.max_byte}"
# max_fs_stat_files = "${var.max_files}"
}
resource "oci_file_storage_export" "my_export_fs1_mt1" {
#Required
export_set_id = "${oci_file_storage_export_set.my_export_set_1.id}"
file_system_id = "${oci_file_storage_file_system.my_fs_1.id}"
path = "${var.export_path_fs1_mt1}"

export_options = [
{
source = "${var.export_read_write_access_source}"
access = "READ_WRITE"
identity_squash = "NONE"
require_privileged_source_port = false
},
{
source = "${var.export_read_only_access_source}"
access = "READ_ONLY"
identity_squash = "ALL"
require_privileged_source_port = false
},
]
}
```

For the Terraform provider to deploy the load balancer, you need to name the load balancer and define the certificates. For more information about the components of the load balancer, see the [documentation](#).

```
variable "load_balancer_name" {
default = "LB01"
}

variable "ca_cert1" {
default = "-----BEGIN CERTIFICATE-----<< Insert CA Certificate >>-----END CERTIFICATE-----"
}
variable "pvt-key1" {
default = "-----BEGIN RSA PRIVATE KEY-----<< Insert Private Key>>-----END RSA PRIVATE KEY-----"
}
variable "pub-cert1" {
default = "-----BEGIN CERTIFICATE-----<<Insert Public Certificate>>-----END CERTIFICATE-----"
}
```

Building the load balancer with the Terraform provider also requires the certificate information and paths for the route.

```
resource "oci_load_balancer_certificate" "lb-cert1" {
load_balancer_id = "${oci_load_balancer.lb1.id}"
ca_certificate = "${var.ca_cert1}"
certificate_name = "certificate1"
private_key = "${var.pvt-key1}"
```

ORACLE

```
public_certificate = "${var.pub-cert1}"

lifecycle {
create_before_destroy = true
}
}

resource "oci_load_balancer_path_route_set" "test_path_route_set" {
#Required
load_balancer_id = "${oci_load_balancer.lb1.id}"
name = "pr-set1"

path_routes {
#Required
backend_set_name = "${oci_load_balancer_backend_set.lb-bes1.name}"
path = "/example/video/123"

path_match_type {
#Required
match_type = "EXACT_MATCH"
}
}
}
```

The load balancer creates the [necessary listeners for the web server hosts](#).

```
resource "oci_load_balancer_hostname" "test_hostname1" {
#Required
hostname = "${var.hostname_1}"
load_balancer_id = "${oci_load_balancer.lb1.id}"
name = "${var.hostname_1}"
}

resource "oci_load_balancer_hostname" "test_hostname2" {
#Required
hostname = "${var.hostname_2}"
load_balancer_id = "${oci_load_balancer.lb1.id}"
name = "${var.hostname_2}"
}

resource "oci_load_balancer_listener" "lb-listener1" {
load_balancer_id = "${oci_load_balancer.lb1.id}"
name = "http"
default_backend_set_name = "${oci_load_balancer_backend_set.lb-bes1.name}"
hostname_names = ["${oci_load_balancer_hostname.test_hostname1.name}",
"${oci_load_balancer_hostname.test_hostname2.name}"]
port = 80
protocol = "HTTP"
rule_set_names = ["${oci_load_balancer_rule_set.test_rule_set.name}"]

connection_configuration {
idle_timeout_in_seconds = "2"
}
}
```

ORACLE

```
resource "oci_load_balancer_listener" "lb-listener2" {
load_balancer_id = "${oci_load_balancer.lb1.id}"
name = "https"
default_backend_set_name = "${oci_load_balancer_backend_set.lb-bes1.name}"
port = 443
protocol = "HTTP"

ssl_configuration {
certificate_name = "${oci_load_balancer_certificate.lb-cert1.certificate_name}"
verify_peer_certificate = false
}
}

resource "oci_load_balancer_backend" "lb-be1" {
load_balancer_id = "${oci_load_balancer.lb1.id}"
backendset_name = "${oci_load_balancer_backend_set.lb-bes1.name}"
ip_address = "${oci_core_instance.instance1.private_ip}"
port = 80
backup = false
drain = false
offline = false
weight = 1
}

resource "oci_load_balancer_backend" "lb-be2" {
load_balancer_id = "${oci_load_balancer.lb1.id}"
backendset_name = "${oci_load_balancer_backend_set.lb-bes1.name}"
ip_address = "${oci_core_instance.instance2.private_ip}"
port = 80
backup = false
drain = false
offline = false
weight = 1
}

load_balancer_id = "${oci_load_balancer.lb1.id}"
name = "example_rule_set_name"
}

output "lb_public_ip" {
value = ["${oci_load_balancer.lb1.ip_addresses}"]
}
```

This Terraform code creates a VCN, a regional subnet, two Windows Server 2016 servers, an NFS file system, and a load balancer. After the environment has been installed, ensure that the web server application and the file system are connected correctly and that the load balancer is in a healthy state. To ensure proper end-to-end function, test the connection of the clients through the load balancer IP address.

ORACLE

Conclusion

To deploy highly available applications, use multiple fault domains. Although fault domains don't provide the continuous availability of separate availability domains, they ensure highly available applications within a low network latent compute environment. This simple demonstration of a high-availability application provides the fundamentals of a regional subnet that services applications in separate fault domains and a load balancer that resides within the availability domain to provide a single point of access for clients to access the application.

Resources

Oracle Cloud Infrastructure documentation

- [Creating Active Directory Domain Services in Oracle Cloud Infrastructure](#) (technical paper)

- [Securing Networking: VCN, Load Balancers, and DNS](#)

- [Compute Shapes](#)

- [Terraform Provider](#)

- [Terraform examples](#)

- [OCI Bastion service](#)

Microsoft documentation

- [Internet Information Services (IIS)](#)

- [Network File System overview](#)

- [NFS PowerShell commands](#)

ORACLE

Appendix A: Variables.tf

Note: The Terraform deployment code provided in this appendix was written using Terraform version 0.10.

```
/* Variables for the Terraform
Need to include Instance, Compartment, and User OCIDs
https://docs.cloud.oracle.com/iaas/Content/API/Concepts/apisigningkey.htm
*/
variable compartment_id {
default = "<< Insert Compartment OCID >>"
}
variable tenancy_ocid {
default = "<< Insert Tenancy OCID >>"
}
variable user_ocid {
default = "<< Insert User OCID >>"
}
/* Secrets information
*/
variable fingerprint {
default = "<<Insert Finger Print>>"
}
variable private_key_path {
default = "<< Insert Private Key path>>.pem"
}
/* Add the Region you want to create the environment:
*/
variable region {
default = "<< Insert Region >>"
}
/* Current Availability Domains

*/
variable availability_domain {
default = "<< Insert Availability Domain >>"
}
variable instance_fault_domain_1 {
default = "FAULT-DOMAIN-1"
}
variable instance_fault_domain_2 {
default = "FAULT-DOMAIN-2"
}
/* Network component name
*/
variable vcn_name {
default = "Webvcn01"
}
variable subnet_name {
default = "Websubnet01"
}
/* Instance Hostnames
*/
variable hostname_1 {
```

ORACLE

```
default = "beinstance1"
}
variable hostname_2 {
default = "beinstance2"
}
/*Virtual Machine Shapes
*/
variable instance_shape {
default = "VM.Standard2.1"
}
/* Image information
See https://docs.us-phoenix-1.oraclecloud.com/images/ or https://docs.cloud.oracle.com/iaas/images/
Oracle-provided image "Windows-Server-2016-Standard-Edition-VM-Gen2-2019.03.14-0"
*/
/*Windows Server 2016
*/
variable "instance_image_ocid" {
type = "map"
default = {
us-phoenix-1 = "ocid1.image.oc1.phx.aaaaaaaatsanuoggs6vzjwhtbmzficdr5364glhroki2o6wp74jva73ldcnq"
us-ashburn-1 = "ocid1.image.oc1.iad.aaaaaaaaq3l2t5p3i2ai6wkwqvi6fltnew4ctxih54kd5fukh4jehe45mnfq"
eu-frankfurt-1 = "ocid1.image.oc1.eu-frankfurt-
1.aaaaaaaaz7zbcqrwkjvi7otldol6ael4qgb56mtvpu3favkwr5einc7zxycq"
uk-london-1 = "ocid1.image.oc1.uk-london-
1.aaaaaaaa6jg4djqfofy2xa44ekcsbd6folvqsgoak4jjdrli2qbtwvtqmlva"
ca-toronto-1 = "ocid1.image.oc1.ca-toronto-
1.aaaaaaaaadtsgda5axsza4nggqpxzc6lymtap3bxi5x7zjl3ohr32t4asnqa"
ap-seoul-1 = "ocid1.image.oc1.ap-seoul-
1.aaaaaaaafcxeuvhwikas37miuycyqsrnoxmng44dld2iaxtuetxm3bak7byq"
ap-tokyo-1 = "ocid1.image.oc1.ap-tokyo-
1.aaaaaaaaivx5l3xcvtnwyutsg3bo3sg4scw2yjchkoodxrjszak3mctxjq4q"
}
}
/* Load Balancer information

*/
variable "load_balancer_name" {
default = "LB01"
}
variable "ca_cert1" {
default = "-----BEGIN CERTIFICATE-----<< Insert CA Certificate >>-----END CERTIFICATE-----"
}
variable "pvt-key1" {
default = "-----BEGIN RSA PRIVATE KEY-----<<Insert Private Key>>-----END RSA PRIVATE KEY-----"
}
variable "pub-cert1" {
default = "-----BEGIN CERTIFICATE-----<< Insert Public Certificate >>-----END CERTIFICATE-----"
}
/* File System information
*/
variable "mount_target_1_display_name" {
default = "NewFileSystem"
}
variable "file_system_1_display_name" {
default = "NewFileSystem1"
```

ORACLE

```
}
variable "export_set_name_1" {
default = "ExportedNFS"
}
variable "export_path_fs1_mt1" {
default = "/Newfilesystem1"
}
variable "export_read_write_access_source" {
default = "10.1.0.0/16"
}
variable "export_read_only_access_source" {
default = "0.0.0.0/0"
}
variable "max_byte" {
default = 23843202333
}

variable "max_files" {
default = 223442
}
```

Appendix B: Main.tf

Note: The Terraform deployment code provided in this appendix was written using Terraform version 0.10.

```
// Copyright (c) 2019, Oracle and/or its affiliates. All rights reserved.
/*
* This example demonstrates round robin load balancing behavior by creating two instances, a
configured
* vcn and a load balancer. The public IP of the load balancer is outputted after a successful run,
curl
* this address to see the hostname change as different instances handle the request.
*
* NOTE: The https listener is included for completeness but should not be expected to work,
* it uses dummy certs.
*/

provider "oci" {
version = ">= 3.0"
tenancy_ocid = "${var.tenancy_ocid}"
user_ocid = "${var.user_ocid}"
fingerprint = "${var.fingerprint}"
private_key_path = "${var.private_key_path}"
region = "${var.region}"
}

/* Network */

resource "oci_core_virtual_network" "vcn1" {
cidr_block = "10.1.0.0/16"
compartment_id = "${var.compartment_id}"
display_name = "${var.vcn_name}"
dns_label = "${var.vcn_name}"
```

ORACLE

```
}

resource "oci_core_subnet" "subnet1" {
availability_domain = "${var.availability_domain}"
cidr_block = "10.1.20.0/24"
display_name = "${var.subnet_name}"
dns_label = "${var.subnet_name}"
security_list_ids = ["${oci_core_security_list.securitylist1.id}"]
compartment_id = "${var.compartment_id}"
vcn_id = "${oci_core_virtual_network.vcn1.id}"
route_table_id = "${oci_core_route_table.routetable1.id}"
dhcp_options_id = "${oci_core_virtual_network.vcn1.default_dhcp_options_id}"

provisioner "local-exec" {
command = "sleep 5"
}
}

resource "oci_core_internet_gateway" "internetgateway1" {
compartment_id = "${var.compartment_id}"
display_name = "internetgateway1"
vcn_id = "${oci_core_virtual_network.vcn1.id}"
}

resource "oci_core_route_table" "routetable1" {
compartment_id = "${var.compartment_id}"
vcn_id = "${oci_core_virtual_network.vcn1.id}"
display_name = "routetable1"

route_rules {
destination = "0.0.0.0/0"
destination_type = "CIDR_BLOCK"
network_entity_id = "${oci_core_internet_gateway.internetgateway1.id}"
}
}

resource "oci_core_security_list" "securitylist1" {
display_name = "public"
compartment_id = "${oci_core_virtual_network.vcn1.compartment_id}"
vcn_id = "${oci_core_virtual_network.vcn1.id}"

egress_security_rules = [{
protocol = "all"
destination = "0.0.0.0/0"
}]

ingress_security_rules = [
{
protocol = "6"
source = "0.0.0.0/0"

tcp_options {
"min" = 80
"max" = 80
}
```

ORACLE

```
        },
        {

        protocol = "6"
        source = "0.0.0.0/0"

        tcp_options {
        "min" = 443
        "max" = 443
        }
        },
        {
        protocol = "6"
        source = "0.0.0.0/0"
        tcp_options {
        "min" = 3389
        "max" = 3389
        }
        },
        {
        protocol = "6"
        source = "0.0.0.0/16"
        tcp_options {
        "min" = 2048
        "max" = 2050
        }
        },
        {
        protocol = "6"
        source = "0.0.0.0/16"
        tcp_options {
        "min" = 111
        "max" = 111
        }
        },
        ]
        }
        /*
        Instance Creation
        */

        resource "oci_core_instance" "instance1" {
        availability_domain = "${var.availability_domain}"
        fault_domain = "${var.instance_fault_domain_1}"
        compartment_id = "${var.compartment_id}"
        display_name = "${var.hostname_1}"
        shape = "${var.instance_shape}"
        subnet_id = "${oci_core_subnet.subnet1.id}"
        hostname_label = "${var.hostname_1}"

        metadata {
        user_data = "${base64encode(file("/Users/jsparker/Desktop/Code/HASingleADReg/lb/bootvars.ps1"))}"
        }
```

ORACLE

```
source_details {
source_type = "image"
source_id = "${var.instance_image_ocid[var.region]}"
}
}

resource "oci_core_instance" "instance2" {
availability_domain = "${var.availability_domain}"
fault_domain = "${var.instance_fault_domain_2}"
compartment_id = "${var.compartment_id}"
display_name = "${var.hostname_2}"
shape = "${var.instance_shape}"
subnet_id = "${oci_core_subnet.subnet1.id}"
hostname_label = "${var.hostname_2}"

metadata {
user_data = "${base64encode(file("/Users/jsparker/Desktop/Code/HASingleADReg/lb/bootvars.ps1"))}"
}

source_details {
source_type = "image"
source_id = "${var.instance_image_ocid[var.region]}"
}
}
/*
File system creation
*/
resource "oci_file_storage_file_system" "my_fs_1" {
#Required
availability_domain = "${var.availability_domain}"
compartment_id = "${var.compartment_id}"

#Optional
display_name = "${var.file_system_1_display_name}"
}

resource "oci_file_storage_mount_target" "my_mount_target_1" {
#Required
availability_domain = "${var.availability_domain}"
compartment_id = "${var.compartment_id}"
subnet_id = "${oci_core_subnet.subnet1.id}"

#Optional
display_name = "${var.mount_target_1_display_name}"
}

resource "oci_file_storage_export_set" "my_export_set_1" {
# Required
mount_target_id = "${oci_file_storage_mount_target.my_mount_target_1.id}"

# Optional
display_name = "${var.export_set_name_1}"
# max_fs_stat_bytes = "${var.max_byte}"
# max_fs_stat_files = "${var.max_files}"
}
```

ORACLE

```
resource "oci_file_storage_export" "my_export_fs1_mt1" {
#Required
export_set_id = "${oci_file_storage_export_set.my_export_set_1.id}"
file_system_id = "${oci_file_storage_file_system.my_fs_1.id}"
path = "${var.export_path_fs1_mt1}"

export_options = [
{
source = "${var.export_read_write_access_source}"
access = "READ_WRITE"
identity_squash = "NONE"
require_privileged_source_port = false
},
{
source = "${var.export_read_only_access_source}"
access = "READ_ONLY"
identity_squash = "ALL"
require_privileged_source_port = false
},
]
}

/* Load Balancer */

resource "oci_load_balancer" "lb1" {
shape = "100Mbps"
compartment_id = "${var.compartment_id}"

subnet_ids = [
"${oci_core_subnet.subnet1.id}",
]
is_private = "TRUE"
display_name = "${var.load_balancer_name}"
}

resource "oci_load_balancer_backend_set" "lb-bes1" {
name = "lb-bes1"
load_balancer_id = "${oci_load_balancer.lb1.id}"
policy = "ROUND_ROBIN"

health_checker {
port = "80"
protocol = "HTTP"
response_body_regex = ".*"
url_path = "/"
}
}

resource "oci_load_balancer_certificate" "lb-cert1" {
load_balancer_id = "${oci_load_balancer.lb1.id}"
ca_certificate = "${var.ca_cert1}"
certificate_name = "certificate1"
private_key = "${var.pvt-key1}"
public_certificate = "${var.pub-cert1}"
```

ORACLE

```
lifecycle {
create_before_destroy = true
}
}

resource "oci_load_balancer_path_route_set" "test_path_route_set" {
#Required
load_balancer_id = "${oci_load_balancer.lb1.id}"
name = "pr-set1"

path_routes {
#Required
backend_set_name = "${oci_load_balancer_backend_set.lb-bes1.name}"
path = "/example/video/123"

path_match_type {
#Required
match_type = "EXACT_MATCH"
}
}
}

resource "oci_load_balancer_hostname" "test_hostname1" {
#Required
hostname = "${var.hostname_1}"
load_balancer_id = "${oci_load_balancer.lb1.id}"
name = "${var.hostname_1}"
}

resource "oci_load_balancer_hostname" "test_hostname2" {
#Required
hostname = "${var.hostname_2}"
load_balancer_id = "${oci_load_balancer.lb1.id}"
name = "${var.hostname_2}"
}

resource "oci_load_balancer_listener" "lb-listener1" {
load_balancer_id = "${oci_load_balancer.lb1.id}"
name = "http"
default_backend_set_name = "${oci_load_balancer_backend_set.lb-bes1.name}"
hostname_names = ["${oci_load_balancer_hostname.test_hostname1.name}",
"${oci_load_balancer_hostname.test_hostname2.name}"]
port = 80
protocol = "HTTP"
rule_set_names = ["${oci_load_balancer_rule_set.test_rule_set.name}"]

connection_configuration {
idle_timeout_in_seconds = "2"
}
}

resource "oci_load_balancer_listener" "lb-listener2" {
load_balancer_id = "${oci_load_balancer.lb1.id}"
name = "https"
default_backend_set_name = "${oci_load_balancer_backend_set.lb-bes1.name}"
```

ORACLE

```
port = 443
protocol = "HTTP"

ssl_configuration {
certificate_name = "${oci_load_balancer_certificate.lb-cert1.certificate_name}"
verify_peer_certificate = false
}
}

resource "oci_load_balancer_backend" "lb-be1" {
load_balancer_id = "${oci_load_balancer.lb1.id}"
backendset_name = "${oci_load_balancer_backend_set.lb-bes1.name}"
ip_address = "${oci_core_instance.instance1.private_ip}"
port = 80
backup = false
drain = false
offline = false
weight = 1
}

resource "oci_load_balancer_backend" "lb-be2" {
load_balancer_id = "${oci_load_balancer.lb1.id}"
backendset_name = "${oci_load_balancer_backend_set.lb-bes1.name}"
ip_address = "${oci_core_instance.instance2.private_ip}"
port = 80
backup = false
drain = false
offline = false
weight = 1
}

resource "oci_load_balancer_rule_set" "test_rule_set" {
items {
action = "ADD_HTTP_REQUEST_HEADER"
header = "example_header_name"
value = "example_header_value"
}

load_balancer_id = "${oci_load_balancer.lb1.id}"
name = "example_rule_set_name"
}

output "lb_public_ip" {
value = ["${oci_load_balancer.lb1.ip_addresses}"]
}
```

ORACLE

Appendix C: Bootvars.ps1

Note: The Terraform deployment code provided in this appendix was written using Terraform version 0.10.

```
#ps1_sysnative
Try
{
Start-Transcript -Path "C:\DomainJoin\stage1.txt"
Add-WindowsFeature Web-Mgmt-Tools, Web-Server
Install-WindowsFeature -Name NFS-Client
Import-Module IISAdministration
Import-Module WebAdministration
$IISFeatures = "Web-WebServer",
"Web-Common-Http",
"Web-Default-Doc",
"Web-Dir-Browsing",
"Web-Http-Errors",
"Web-Static-Content",
"Web-Http-Redirect",
"Web-Health",
"Web-Http-Logging",
"Web-Custom-Logging",
"Web-Log-Libraries",
"Web-ODBC-Logging",
"Web-Request-Monitor",
"Web-Http-Tracing",
"Web-Performance",
"Web-Stat-Compression",
"Web-Security",
"Web-Filtering",
"Web-Basic-Auth",
"Web-Client-Auth",
"Web-Digest-Auth",
"Web-Cert-Auth",
"Web-IP-Security",
"Web-Windows-Auth",
"Web-App-Dev",
"Web-Net-Ext",
"Web-Net-Ext45",
"Web-Asp-Net",
"Web-Asp-Net45",
"Web-ISAPI-Ext",
"Web-ISAPI-Filter",
"Web-Mgmt-Tools",
"Web-Mgmt-Console"
Install-WindowsFeature -Name $IISFeatures
Remove-Website -Name "Default Web Site"
$defaultAppPools = @(".NET v2.0",".NET v2.0 Classic",".NET v4.5",".NET v4.5 Classic","Classic .NET
AppPool","DefaultAppPool")
Foreach ($defaultAppPool in $defaultAppPools)
{
IF (Test-path "IIS:\AppPools\$defaultAppPool"){Remove-WebAppPool -name $DefaultAppPool}
}
```

ORACLE

```
$NewFolders = "inetpub", "inetpub\apps", "logs"
$NewFolders | ForEach-Object {New-Item C:\$_ -type directory}
Set-WebConfigurationProperty "/system.applicationHost/sites/siteDefaults" -name logfile.directory -
value C:\logsx


} Catch {
Write-Host $_
} Finally {
Stop-Transcript
}
```

Connect with us

Call **+1.800.ORACLE1** or visit **oracle.com**. Outside North America, find local offices at **oracle.com/contact**.

blogs.oracle.com   facebook.com/oracle   twitter.com/oracle

ORACLE