ORACLE

OCI Open Landing Zone Blueprint

A Blueprint to Simplify the Onboarding of Organizations, Business Units, and Subsidiaries into OCI.

September 2023
Version 1.0.1
Public

Table of contents

Table of figures

Version Control

General

VERSION	AUTHORS	DATE	COMMENTS
V1.0.1	JP, OH	14/09/2023	Minor updates and corrections.
V1.0.0	JP, CT	28/07/2023	Updated repository with Runtime View, configurations, and code artefacts
V0.17	JP	21/07/2023	Updated links to assets
v0.16	JP, OH, PK, CT, PJ, PA	10/06/2023	Draft version.

Authors

ID	NAME	ROLE
CT	Cosmin Tudor	Cloud Operations Specialist | EMEA Technology Engineering
JP	José Palmeiro	Head of Landing Zones and Operations | EMEA Technology Engineering
OH	Olaf Heimburger	Cloud Security Advisor | EMEA Technology Engineering
PA	Pablo Alonso	Cloud Operations Specialist | EMEA Technology Engineering
PJ	Paola Juarez	Cloud Operations Specialist | EMEA Technology Engineering
PK	Par Kansala	Multi-cloud Specialist | EMEA Technology Engineering

References

REF. ID	NAME	VERSION	COMMENTS
REF.00	OCI Open LZ	V1.0.0	The OCI Open LZ and all its artifacts can be found here.
REF.01	Draw.io	--	Architecture diagrams published on the OCI Open LZ Git.
REF.02	Standard Landing Zones	--	Oracle EMEA Landing Zone Framework assets.
REF.03	CIS Landing Zone	2.x	OCI secure landing zone that validates the CIS Benchmark
REF.04	OELZ	2.x	Oracle Enterprise Landing Zone
REF.05	Tailored Landing Zones	--	Oracle EMEA Landing Zone Framework assets.
REF.06	Resource Naming Conventions	--	Published on the public OCI Landing Zones Framework.
REF.07	Budgets and Tagging	--	Published on the public OCI Landing Zones Framework.
REF.08	User Identity Management	--	Published on the public OCI Landing Zones Framework.
REF.09	CIS Landing Zone IAM	Git	Git Repository for IAM CIS LZ Enhanced Modules
REF.10	CIS Landing Zone Network	Git	Git Repository for Network Modules CIS LZ Enhanced Modules
REF.11	OCI Open LZ Runtime View	Git	This chapter is OCI Open LZ Git Repository.
REF.12	CIS LZ Enhanced Terraform Modules	--	CIS Landing Zone Enhanced Modules blog post.

Introduction

Purpose

The purpose of this document is to:

- Provide a landing zone design ready to **onboard an enterprise organization** and its functional divisions - identified as **operating entities (OE)** with their teams, departments, and projects.
- Provide a cloud-native **operating model** to simplify and scale **day 2 operations**.
- **Enable customers, partners,** and the **general IT community** to **create their own landing zones** with lower efforts through a comprehensive Oracle Cloud Infrastructure (OCI) reference architecture. To support this objective, all the architecture diagrams are provided in a reusable format [REF.01].
- Provide **tailoring guidelines** to help adjust the model. This asset can be used directly, tailored, or used as inspiration to create a new one - as it is not a prescribed solution.

Vision

The **OCI Open LZ,** which is the short name for **Operating Entities Landing Zone,** is a secure cloud environment, designed with best practices to simplify the **onboarding** of organizational **operating entities** (e.g., Line-of-Business, OpCos, Subsidiaries, Departments, etc.) and enable the **continuous operations** of their cloud resources.

The OCI Open LZ is also a **blueprint** for creating new OCI landing zone, presented with a repeatable and guided design approach, that can be used as a reference model for different business needs. Find below some of the OCI Open LZ key characteristics.

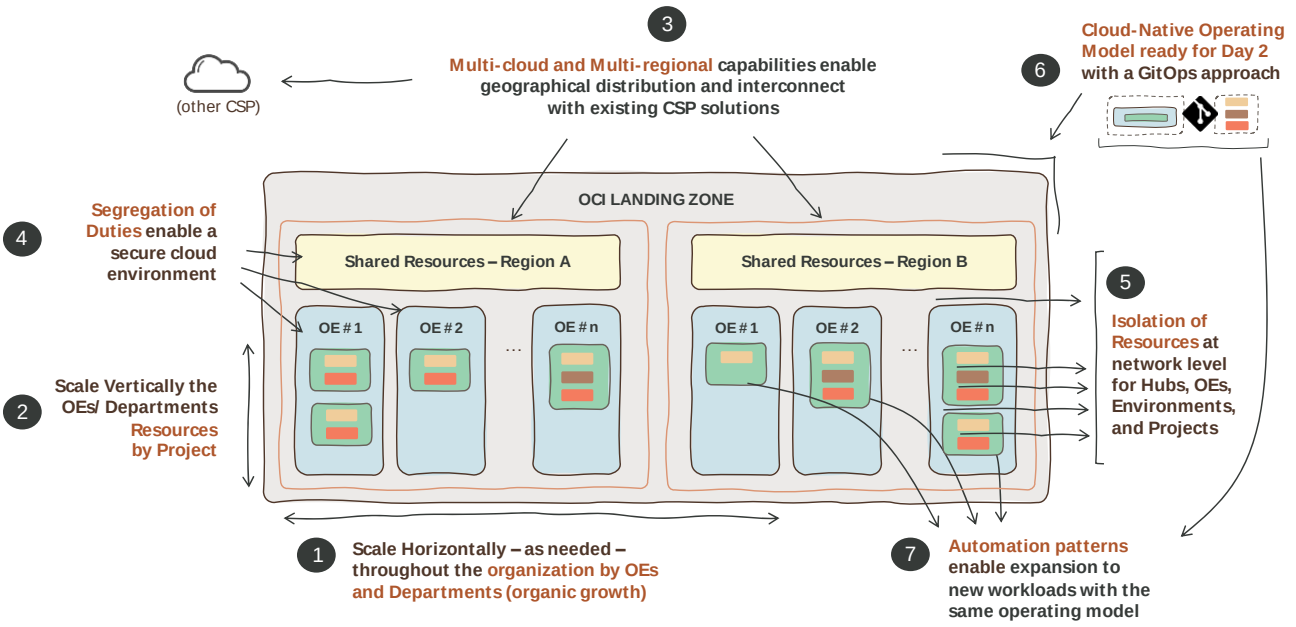


Figure 1 OCI Open LZ Characteristics.

#	CHARACTERISTIC	DESCRIPTION
1	**Enterprise Organization / Horizontal Scaling**	The OCI Open LZ is designed to simplify the OCI onboarding of an enterprise organization, and its Operating Entities (OE) with their teams, departments, and projects.
2	**Project-Driven Structure / Vertical Scaling**	The OCI Open LZ is ready to onboard several projects and environments, as a base for their related workloads.

3	**Multi-Cloud / Multi-Region**	The OCI Open LZ can exist in several OCI regions and be connected to other landing zones in other CSPs.
4	**Segregation of Duties**	All OCI Open LZ elements can be segregated in terms of identity and access management. There is an enterprise segregation of resources for shared elements and OE-dedicated elements. An OE can have several OE departments or project owners, which will be responsible for their resources, organized by projects.
5	**Isolation of Resources**	The OCI Open LZ has an isolation of resources at the network level. The network structure is organized by OE, into environments (production and non-production) and resources in those environments are isolated at the project level, where each project layer has its security posture.
6	**Cloud Native Operating Model**	The OCI Open LZ can be operated with a complete GitOps operating model on day 2, using control version repositories as the single source of truth for operations and code. The OCI Open LZ uses Infrastructure as Code (IaC) and IaC configurations on git-versioned repositories.
7	**Automation Patterns**	The OCI Open LZ has a set of operations scenarios for provisioning and changing resources, providing the building blocks to design and automate any other repeatable operations.

Approach and Timeline Considerations

A landing zone can be set up in different ways and can take different amounts of time to implement. There are mainly two types of approaches:

1. **Standard and prescribed approaches** [REF.02], are the recommended starting point and can take hours to days to set up. This option enables quick start cloud adoption with a set of recommended best practices with a prescriptive design. For more details on this type of approach please refer to the CIS Landing Zone [REF.03] or OELZ [REF.04].
2. **Tailored approaches** [REF.05] focus on creating a landing that fits completely the requirements. They normally cover security, network, and operational topics, and can onboard a complete enterprise organization with one cloud operating model. This option is recommended when the standard approach is not enough (e.g., large organizations with fine-tuned security or network requirements, large and heterogeneous workloads landscape with multi-cloud scenarios, etc.) and experience tells us it can take from days to weeks to set up - depending on requirements and team expertise.

The **OCI Open LZ** is an example of the outcome of the latter approach, a tailored landing zone, and one of its purposes is to help reduce the design time, associated cost, and effort. To further support this, all architecture diagrams are provided in a reusable format [REF.00] [REF.01].

Scope and Organization

This OCI Open LZ is presented with several design views built on top of each other, as an incremental and repeatable approach, that can be used and tailored by any customer or partner setting up an OCI landing zone. Each view is explored in a dedicated chapter:

1. The **Functional View** presents the fundamental organizational entities of the OCI Open LZ and how they relate to each other.
2. The **Security View** presents the core building blocks of the tenancy organization and security design.
3. The **Network View**, designed on top of the security, presents how network elements are structured and connected to communicate with each other.
4. The **Operations View** presents the dynamic perspective of the OCI Open LZ**,** proposing a design for provisioning and changing the previous elements with an operating model ready for day 1 and day 2 operations.
5. The **Runtime View** presents the OCI Open LZ operations artifacts to demonstrate how day 2 operations can run.

The OCI Open LZ and its views provide a consistent design to simplify the onboarding of OCI with an existing blueprint, that can be changed and tailored toward different objectives. Note the order in which these views are presented is itself a best practice, and it's crucial to reproduce the approach with lower efforts and less rework. Therefore, changing security elements will impact the network elements, and any change in these will impact operations. Any change operations will naturally impact the runtime of the OCI Open LZ.

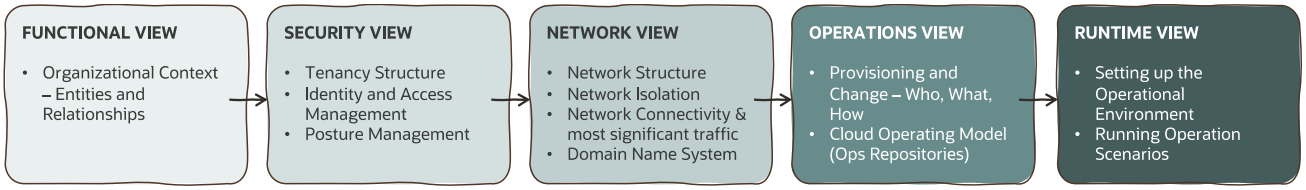


Figure 2 Journey to Onboard OCI.

While the previous diagram presents the recommended steps of the journey to create an advanced landing zone, such as the OCI Open LZ, the diagram below presents the layered perspective to emphasize that each inner layers impact the design or considerations of any outer layer.

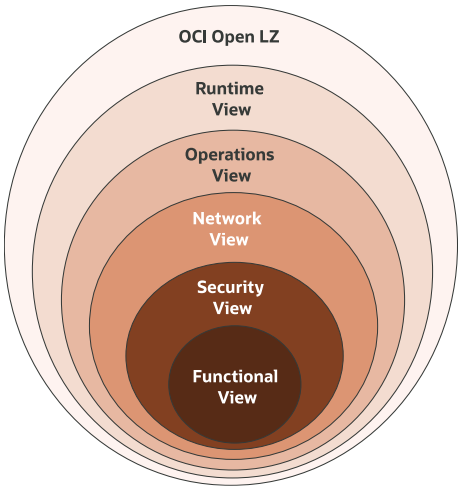


Figure 3 OCI Open LZ Layers and Chapters.

It's out of scope of the OCI Open LZ the workload elements as they can follow the same reference design and operating model, and the monitoring view, which will be added in the future as an incremental update.

Before proceeding, it's highly recommended OCI foundational knowledge on its core services and resources, such as Compartments, Groups, Policies, DRG, VCNs, Subnets, Route tables, Security Lists, Network Security Groups, among others. For the operations view it's recommended intermediate knowledge on version control systems, pipelines, and infrastructure-as-code (IaC).

Functional View

The objective of this chapter is to:

- Present the key organizational entities of the OCI Open LZ and how they relate to each other.
- Provide tailoring guidelines to help adjust the model.

Entities and Relationships

This Landing Zone pattern is driven by **Operating Entities (OE)**, the higher unit of landing zone resource aggregation. This resource aggregation requires a shared operating model with several levels of responsibility:

1. **Central IT Responsibility (Central Operations – 1 Team)**: these resources will be operated by a Central IT Operations team which will be completely responsible for the landing zone shared elements.
2. **Operating Entities Responsibility (OE Operations – N Teams)**: these resources will be operated by each OE, including onboarding their departments and projects. A project will contain a set of OCI resources including applications, databases, and related infrastructure.

Find below the entities relationship diagram (ERD) associated with this landing zone model.

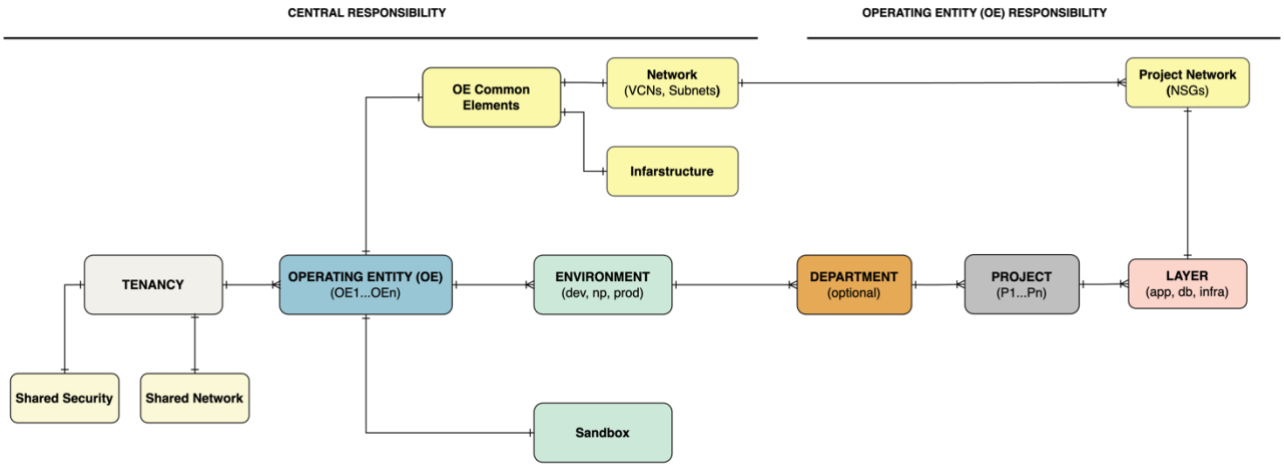


Figure 4 Functional View – Entities and Relationships Diagram.

The following table describes each entity, its level of deepness in the tenancy along with its cardinality.

ELEMENTS	DESCRIPTION	RESPONSIBILITY	TARGET RESOURCES	#
ROOT	The root of the OCI Tenancy.	Central Ops	Groups, Policies.	1
SHARED SERVICES	The parent company, normally the customer IT organization, will operate the landing zone shared elements across Operating Entities, providing network and security services to all OEs.	Central Ops	VCN, Subnets, NSG, Firewalls, DNS. Security resources like Vaults, Logs, and Integrations (e.g., SIEM, Monitoring).	1
OPERATING ENTITY (OE)	The Operating Entity will leverage OCI and have its standard area for using OCI resources. An OE can be a LoB, a Subsidiary, an OpCo, etc. There can be N OEs in the tenancy. This capability can be viewed as a horizontal	Central Ops	Access management and policy enforcement area.	1..N

	scaling to support organic growth. An OE can be seen also as a sub-tenant.			
OE COMMON ELEMENTS	This element contains the common resources of an OE:	Shared	Access management and policy enforcement area, and Core Network shared elements.	1
	• **Common Security Resources** for the OE	Central Ops	Logs, Integrations (e.g., SIEM, Monitoring), etc.	1
	• **Common Network Resources** for the OE	Central Ops	VCN, Subnets, RT, Gateways, Firewalls, DNS, Security Lists (SLs)	1
	• **OE Project Network Security** elements.	OE Ops	Network Security Groups (NSGs)	1
	• **Common Infrastructure Resources** for the OE	Central Ops	E.g., ExaCS	1
ENVIRONMENT	An OE has usually several environments:	Central Ops	Access management and policy enforcement area.	3
	• **Sandbox:** Used for exploration mode, the OE can test and use OCI in a sandbox area to deploy PoCs projects.	OE Ops	PoC Compute, Storage, Databases, etc.	1
	• **Development:** Used when for OE development environment	Central Ops	Access management and policy enforcement area.	1
	• **Non-Production:** Used when the OE promotes projects to non-production environments.	Central Ops	Access management and policy enforcement area.	1
	• **Production:** Used for the OE projects in the production environment.	Central Ops	Access management and policy enforcement area.	1
DEPARTMENT	OE department. One OE can hold several departments.	OE Ops	Access management and policy enforcement area	0..N
PROJECT ENVIRONMENT	A project is a set of related resources used toward the same goal. It can contain one or several applications, databases, and related infrastructure. One OE/department can have several projects associated with environments (e.g., dev, int, uat, prod, dr).	OE Ops	Access management and policy enforcement area	1..N
PROJECT LAYER	**Project Applications** Layer.	OE Ops	Compute resources.	1
	Project Database Layer.	OE Ops	Databases (e.g., DBCS, ATP-S)	1
	Project Infrastructure Layer.	OE Ops	Storage, Support Compute, etc.	1

Context Tailoring

The functional context presented above can be simplified according to requirements. Ultimately this landing zone can be simplified into a project-driven landing zone, having only the **project** layers directly at level L1, or at L2 - where the **environments** could be value-added to avoid project proliferation without environments segmentation. Find below some rationale behind each decision.

ELEMENTS	TAILORING TYPE	WHY KEEP OR REMOVE
ROOT	MANDATORY	--
SHARED SERVICES	MANDATORY	Shared Services are part of a Landing Zone definition and are owned by the parent company.
OPERATING ENTITY (OE)	RECOMMENDED	This element is key to the pattern. Keep if it's required to separate organizations, their users, permissions, and costs, reflecting the organizational structure. Remove if the landing zone will not reflect the organization structure.
OE COMMON ELEMENTS	OPTIONAL	Keep this element if each OE has dedicated resources that the OE team is responsible of manage, or if in the future an OE might manage. Remove if all shared resources are managed in a centralized way, i.e., by one operations team, and it's not envisioned that an OE can control these in the future. Add a new OE Common Element "Security" entity if the OE has dedicated security resources or integration requirements, not shared with the other OEs.
ENVIRONMENT	RECOMMENDED	In a higher granularity than project environments, environments are normally required to properly operate a landing zone. Adjust the environments to the reality and standards. These environments allow segregation of duties, and isolation of resources, and enable cost management per environment. Remove it if the landing zone has a flat structure driven by projects/applications which has their environments, or if hard segregation between production and non-production resources is not required at a higher level.
DEPARTMENT	OPTIONAL	Keep this element if an OE has several departments and it's needed to separate users, permissions, resources, and costs, under each OE. Optionally in a no-department scenario for an OE, a default department can be created to have a consistent and deterministic way of operating the OCI Open LZ. Remove if the landing zone doesn't need to reflect this intermediate structure or if an OE doesn't have departments.
PROJECT ENVIRONMENT	MANDATORY	This is the lowest level of resource aggregation in a landing zone, it should not be removed.
PROJECT LAYER	RECOMMENDED	Keep this element if the projects have several layers and potentially different teams are responsible for their management. Remove it if there is only one layer per project (e.g., databases).

Security View

A good security design implies a good tenancy design. This chapter contains the following security elements:

- **Tenancy Structure** presents the compartment structure to support resource grouping, separation of duties, and budget control and billing.
- **Identity and Access Management** defines the groups, dynamic groups, and policies for the related duties and compartments.
- **Configuration and Posture Management** describes the additional configurations for native security tooling used in this pattern.
- **Design Considerations** presents guidelines complementing or extending this design.

All the resource names in this chapter follow the Oracle Resource Naming Conventions [REF.06] and serve as a guideline. The customer's resource naming convention should be used, or merged with this proposal, when available.

Tenancy Structure

An OCI tenancy structure has OCI compartments, groups, and policies as core resources. Compartments provide the ability to organize and isolate resources in an OCI tenancy and play an essential role in setting a foundation for deploying workloads. Although they may appear to have the nature of logical grouping of OCI resources, they serve as policy enforcement points, thus they have paramount importance concerning the tenancy's **security**.

Compartments can be deployed according to a functional, operational, or project hierarchy. This allows for maintaining isolation between resources for different roles, functions, and organizational hierarchies. A compartment hierarchy can have up to 6 levels, based on the requirements. Access control is defined by policies, which are associated with compartments.

The compartment structure of the OCI Open Landing Zone is aligned with the functional context, and reflects the following **design principles** regarding segregations of resources and duties:

- **P1:** Segregation between **Shared Resources** and **Operating Entities** elements (e.g., OE01, OE02, etc.)
- **P2:** Segregation among **Operating Entities** and their potential **departments,** reflecting customer organizational structure (e.g., OE01, OE02, etc.)
- **P3:** Segregation of **environments** like Production and Non-production environments
- **P4:** Segregation among **applications/projects** belonging to departments.
- **P5:** Segregation among **application layers** (App, DB, and Infra) belonging to applications/projects.

The diagram below presents the tenancy structure for Level 1 (L1), associated with principle P1 identified above, separating the **shared resources area** (in pale yellow) operated by the parent company, and the N possible **operating entities areas** (in blue tones), that each OE will be responsible for operating their resources.

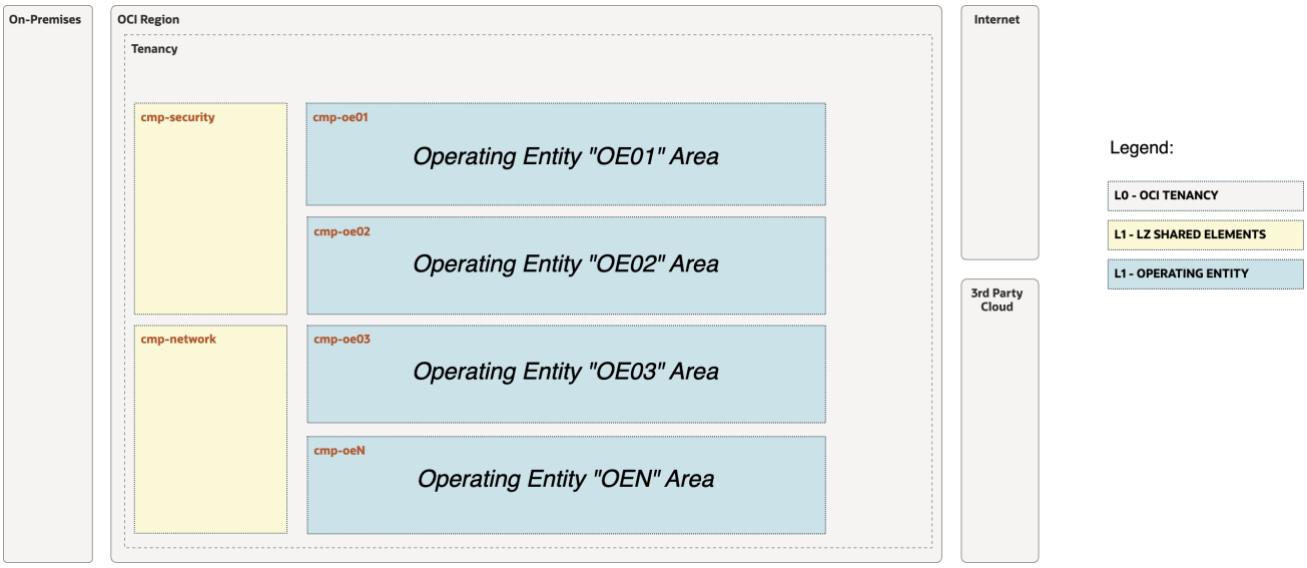


Figure 5 Security View – Tenancy Structure (L0-L1).

Expanding the tenancy structure into Levels 1, 2, and 3 (L1, L2, and L3) as shown in the architecture below. it includes a shared compartment for each OE and is complemented by three types of environments: sandbox, non-production, and production (in green).

Figure 6 Security View – Tenancy Structure (L0-L2).

The diagram below presents a zoom on the tenancy structure for an Operating Entity named "OE01" from L1 to L5 elements. All OEs will have this same structure.

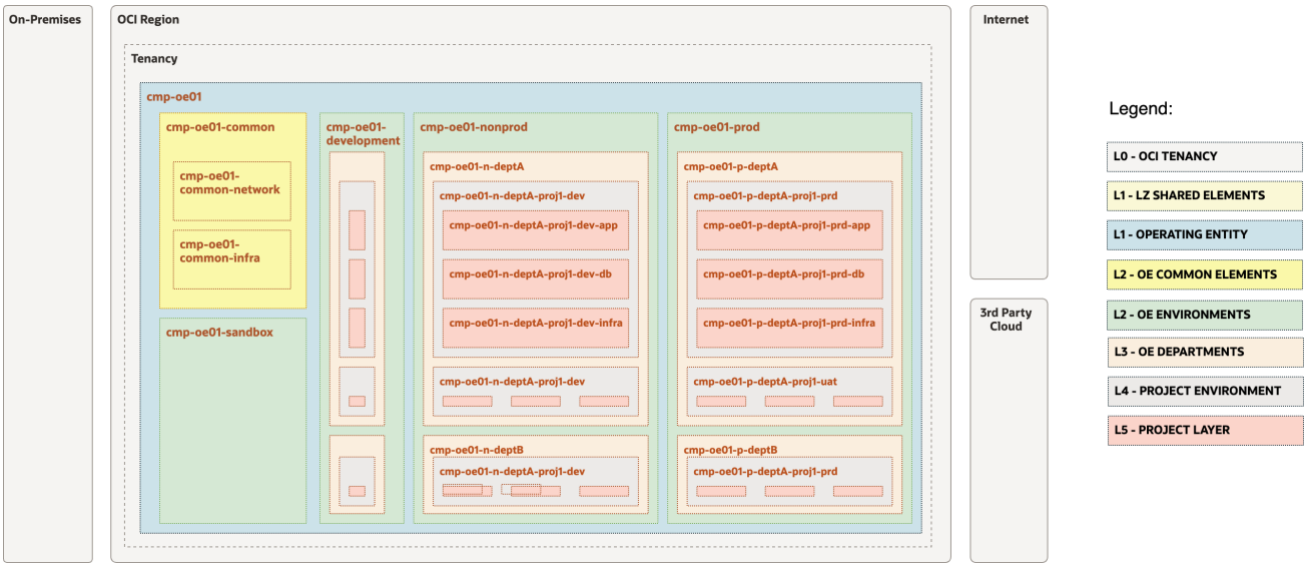


Figure 7 Security View – Tenancy Structure (L0-L5).

The following table provides some details for the compartments presented above, their level of deepness in the tenancy, objectives, and target resources.

SCOPE	ID	COMPARTMENT NAME	L	OBJECTIVE	TARGET RESOURCES
ALL	CMP.00	(root) Tenancy	0	Holds tenancy global resources	Target of Cloud Guard and Vulnerability Scanning; Recipes for Cloud Guard, Security Zones, VSS, Budget Control.
SHARED	CMP.01	cmp-security	1	Support shared central resources associated with security	Vaults, Monitoring, Logging, Audit, Integrations (e.g., SIEM, Monitoring), etc.
	CMP.02	cmp-network	1	Support shared central resources associated with network	All OCI Core Network Elements
OE	CMP.03	cmp-<oe_name>	1	OE dedicated area. All OE resources in the tenancy will be deployed here.	--
	CMP.04	cmp-<oe_name>-common	2	OE common resources area, including security and network.	--
	CMP.05	cmp-<oe_name>-common-infra	3	OE common infrastructure resources.	E.g., Shared ExaCS, OCVS, OKE, etc. No Security resources.
	CMP.06	cmp-<oe_name>-common-network	3	OE common network resources. OE Project security (NSGs)	VCNs, Subnets, RT, Gateways, Firewalls, DNS, SL. NSGs
	CMP.07	cmp-<oe_name>-sandbox	2	OE Area for explorations with support for PoC Projects	PoC Workloads
	CMP.08	cmp-<oe_name>-development	2	OE Development dedicated area to hold related project environments	--
	CMP.09	cmp-<oe_name>-nonprod	2	OE Non-production dedicated area to hold related project environments	--
	CMP.10	cmp-<oe_name>-prod	2	OE Production dedicated area to hold related project environments	--

	CMP.11	cmp-<oe_name>-<environment>-<department>	3	OE Area for an OE Department	--
	CMP.12	cmp-...-<dept>-<project_name>-<project_environment>	4	OE Department area for a project environment	--
	CMP.13	cmp-...-<proj_environment>-app	5	Project environment application layer	Application Workloads
	CMP-14	cmp-...-<proj_environment>-db	5	Project environment Database layer	Database Workloads
	CMP-14	cmp-...-<proj_environment>-infra	5	Project environment Infrastructure layer	Infrastructure support for Workloads

Budget Control and Billing

For budget control and billing two approaches are recommended:

- Tenancy-global to identify unauthenticated usage.
- OCI Open LZ structure specific

While the tenancy-global budget control applies to the overall average consumption, the OCI Open LZ compartment structure enables the creation of budgets to set soft limits on OCI tenancy spending, by OCI Open LZ Shared Elements, OEs, Departments, and Projects. With budget control, the available budget can be controlled using quotas and trigger alarms when unusual costs occur.

Budget control and billing is a global task for every tenancy, and since it's a generic OCI topic not specific to the OCI Open LZ. For additional information see Budgets and Tagging [REF.07]

Identity and Access Management

Identity and Access Management design define the groups and policies defined for the tenancy. A good design includes the segregation of duties and applies to any usage scenario.

As a general principle, groups follow the least privilege principle. However, users can be members (but rarely should be) of multiple groups if needed.

This pattern uses the groups and policies implemented by the CIS Landing Zone (Standard Landing Zone), plus the OE-specific groups and policies.

Operation roles can combine these roles if needed.

Personas

User life cycle management and creation is the whole responsibility of the customer.

There are four type of user personas:

- Emergency Users (Break Glass scenario)
- OCI Administrators
- Cloud Operators
- Workload Users for OS, Database, and Applications

USER PERSONAS	GUIDELINES
EMERGENCY USERS	• For ***Emergency or Break Glass*** scenarios, customers should have up to three (3) dedicated users who are **not federated**, i.e., have different usernames and do not use these accounts daily! • Emergency Users are there to help when nothing else works (like Federation or initial setup). • The life cycle management of emergency users might be a manual process and should be done carefully by the customer.
OCI ADMINISTRATOR	• For regular OCI administrators, it is the customer's responsibility to integrate OCI with the existing (third-party) Identity and Access Management system through SAML 2.0 or OpenID Connect federation and configure user provisioning for joiners, movers, and leavers (using the SCIM protocol). This integration uses the primary Identity Cloud Service or the Default Domain. • All the groups used in the third-party system may be provisioned to OCI and used by the policy statements.
CLOUD OPERATIONS	• Refer to Operations View chapter.
WORKLOAD USERS FOR OS, DATABASE, AND APPLICATION	• Workload Users are all users that do not administer the OCI tenancy. These users **must not** be provisioned to the primary Identity Cloud Service, the Default Domain, or created as Emergency Users. • All Workload Users are expected to be managed in the customer's existing (third-party) Identity and Access Management system. For the provisioning of OCI-based PaaS services a separate Identity Cloud Service or Domain is required. • For a full description see User Identity Management [REF.08]

Groups and Federation

The table below presents a list of **Groups** associated to **Policies** and **Compartments**. Policies will give these groups permission to work on the compartments. The group names are suggestions, and as a general guideline, existing group names or existing naming conventions on the target Identity Provider system can be used.

In terms of **federation**, note that when an OCI **Identity Domain** is federated with a third-party **Identity Provider**, the user footprint will be synchronized from the Identity Provider to the OCI Identity Domain. This **synchronization** will **create existing Identity Provider group** names on the **OCI Identity Domain**.

ID	GROUP NAME	DESCRIPTION	POLICY NAME	TARGET COMPARTMENTS
GRP.00	N/A	Policy for all services	pcy-services	Tenancy
GRP.01	grp-iam-admins	Tenancy global Identity and access management administrator. (Needs to be reviewed for IAM Identity Domains.)	pcy-iam-administration	Tenancy
GRP.02	grp-credential-admins	Tenancy global credential administrator. (Needs to be reviewed for IAM Identity Domains.)	pcy-credential-administration	Tenancy
GRP.03	grp-announcement-readers	Tenancy global readers of OCI monitoring information.	pcy-announcement-read	Tenancy
GRP.04	grp-budget-admins	Tenancy global budget control.	pcy-budget-administration	Tenancy
GRP.05	grp-auditors	Tenancy global read access (for security auditing or health checks)	pcy-auditing	Tenancy
GRP.06	grp-network-admins	Tenancy global and shared network administration group, including common OE network elements.	pcy-networking-administration	Tenancy and cmp-network

GRP.07	grp-security-admins	Tenancy global and shared security administration group.	pcy-security-administration	Tenancy and cmp-security
GRP.OE.01	grp-<oe>-admins	<OE> specific administrator group responsible for creating compartments.	pcy-<oe>-administration	cmp-<oe>
GRP.OE.02	grp-<oe>-network-admins	This group can use the <OE> common networking and manages project NSGs.	pcy-<oe>-network-administration	cmp-<oe>-common-network
GRP.OE.03	grp-<oe>-app-admins	This group is responsible for administrating <OE> related applications, PaaS, etc.	pcy-<oe>-application-administration	cmp-<oe>-<...>-app
GRP.OE.04	grp-<oe>-db-admins	This group is responsible for administrating <OE> related databases.	pcy-<oe>-database-administration	cmp-<oe>-<...>-db

Dynamic Groups

Dynamic Groups define the resources which are allowed to be used by OCI native services and do not require any actual user. The table below is not exhaustive and may be expanded (see 2.4 Design Considerations).

ID	GROUP NAME	DESCRIPTION	POLICY
DGP.01	dgp-os-management	Holds the compartments which contain the VM images to be automatically patched by the OS Management Service.	
DGP.02	dgp-autonomous-db	Holds the compartments of Autonomous DBs which can use Security resources like Vaults and Customer-Managed Keys for encryption.	pcy-dgp-databases
DGP.03	dgp-security-functions	Allows all resources of type fnfunc in the Security compartment, cmp-security.	

Policies

Policies define the permissions assigned to a group. The respective policy statements are implemented using the OCI CIS Landing Zone Terraform module.

ID	GROUP NAME	DESCRIPTION	POLICY NAME	TARGET COMPARTMENTS
POL.00	pcy-services	Allows Cloud Guard to read resources in the tenancy. Allows Vulnerability Scanning to read resources in the tenancy. Allows Data Safe to read resources in the tenancy. Allows OS Management Service to read resources in tenancy.	View	All supported resources in the tenancy.
POL.01	pcy-iam-administration	Allows the users to manage IAM resources in the tenancy.	Manage	IAM resources in the tenancy.
POL.02	pcy-credential-administration	Allows the users to manage user credentials of local users in the tenancy.	Manage	User credentials in the tenancy.
POL.03	pcy-announcement-readers	Allows the users to read OCI service announcements in the tenancy.	Read	Announcement resources in the tenancy.
POL.04	pcy-budget-administration	Allows the users to manage all budget resources in the tenancy.	Manage	All budget resources in the tenancy.
POL.05	pcy-auditing	Allows the users to read all the resources in the tenancy.	Inspect Read	All resources in the tenancy.

POL.06	**pcy-network-administration**	Allows the users to manage all network resources in the compartment. This includes all networking components, such as VCNs, subnets, gateways, virtual circuits, security lists, route tables, etc.	Manage	All network resources.
POL.07	**pcy-security-administration**	Allows the users to manage all security resources in the security compartment.	Manage	All security-related resources.
POL.DGP.01	**pcy-dgp-databases**	Allows the database management processes to use vaults and keys for customer-managed key management available in the cmp-security compartment.	Manage	All databases of the dynamic group.
POL.OE.01	**pcy-<oe>-administration**	Allows the users to manage the compartment structure of the OE.	Manage	Compartment
POL.OE.02	**pcy-<oe>-network-administration**	Allows the users to manage NSGs in the OE network compartment. Uses resources from the shared OE common network compartment.	Use Manage	Network-related resources.
POL.OE.03	**pcy-<oe>-app-administration**	Allows the users to manage application resources in the OE application compartment not managed by the shared network and security compartments. Uses resources from the shared network, the shared security, and the OE network compartment compartments.	Use Manage	Application-related resources
POL.OE.04	**pcy-<oe>-db-administration**	Allows the users to manage database resources in the OE application compartment not managed by the shared network and security compartments. Uses resources from the shared network, the shared security, and the OE network compartment compartments.	Use Manage	Database-related resources.

Posture Management

The tenancy security posture management is a combination of the tenancy security design using compartments, groups, and policies, complemented by the usage of native security services.

For the compartments, groups, and policies refer to the sections above. To complement the security posture management the following OCI native services should be implemented:

- **Cloud Guard**, to examine OCI resources for security weakness related to configuration, and OCI operators and users for risky activities. More details here.
- **Vulnerability Scanning Service**, helps improve the security posture by routinely checking hosts and container images for potential vulnerabilities. The service gives developers, operations, and security administrators comprehensive visibility into misconfigured or vulnerable resources and generates reports with metrics and details about these vulnerabilities including remediation. More details here.
- **OS Management**, manages and monitors updates and patches for the operating system environments, including Oracle Autonomous Linux, and discover and monitor resources on OCI instances. More details here.
- **Data Safe** focuses on the security of data. It provides a complete and integrated set of features for protecting sensitive and regulated data in OCI databases. Features include Security Assessment, User Assessment, Data Discovery, Data Masking, and Activity Auditing. More details here.
- **Budget Control and Quota** should be applied for the whole tenancy.

These services form the foundation (or shared services) for the whole landing zone and are normally set up on the initial steps of the landing zone deployment.

POSTURE MANAGEMENT SCOPE	TARGET COMPARTMENTS	RESOURCES	POSTURE GUIDELINES	SCOPE
Tenancy Global Configuration	(root)	Shared Compartments	Service policy should be enabled	Global
		Groups	See Groups, non-OE groups	Global
		Dynamic Groups	See Dynamic Groups	Global
		Policies	See Policies, non-OE policies	Global
		Cloud Guard	Target will be defined in the root compartment and using all predefined Oracle-managed recipes and the root compartment. Will be enabled using the target.	Global in the Home region
		Vulnerability Scanning	A target will be created in the root compartment. Defines scanning schedule as WEEKLY on SUNDAY. Vulnerability Scanning will be enabled using the target.	Regional
		Security Zones	Creates recipes at the root level for: • SZ-R1: Deny the creation of any network resources. • SZ-R2: Deny the creation of any database and public resources. • SZ-R3: Deny moving data to public resources. Assigns Security Zone recipes to cmp-network and cmp-security	Global
		Data Safe	Data Safe will be enabled	Regional
		OS Management	Enabled by pcy-services and dgp-os-management	Global
		Budget Control and Quotas	Tenancy global budget and quota will be created	Global
Shared Security Compartment	cmp-security	Vaults	Vaults for OEs will be created here. The type of Vaults (shared or virtual private) depends on the usage patterns (if import, export, and replication are required, a virtual private vault can be an option). Pricing of each type must be considered.	Regional
		Security Zones	Security Zone with recipe SZ-R1 assigned to this compartment	Regional
Shared Network compartment	cmp-network	Networks and Logging and Monitoring of Network resources. As defined in the Network View.	Refer to Network Security Posture.	Regional
		Security Zones	Security Zone with recipes SZ-R2, and SZ-R3 assigned to this compartment.	Global
OE Compartment	cmp-<oe>	Security Zones	Use recipes R1, R2, and R3	Global
	cmp-<oe>-common-network	Network Resources	Refer to Network Security Posture.	Regional

Design Considerations

The previous sections presented an OCI cloud-native security design. There are several alternatives to this model, based on possible requirements. Consider the following topics when adjusting the OCI Open LZ.

TOPIC	ID	DECISION	GUIDELINES	TARGET COMPARTMENT
Identity Store	D1	**OCI Administrator user store**	OCI Administrators are using the Default Domain.	root
	D2	**Identity Store for OCI PaaS Services**	OCI PaaS services should not use the Default Domain as their user store. Separate user stores per application and environment type (Production, Non-production, Test, UAT) should be considered to prevent erroneous mixing of users, permission, and side effects because of thrashing (permissions are not predictable because of synchronizations).	cmp-<oe_name>
	D3	**Identity Store for Applications**	Workload applications should not use the Default Domain as their user store. Separate user stores per application and environment type (Production, Non-production, Test, UAT) should be considered to prevent erroneous mixing of users, permission, and side effects because of thrashing (permissions are not predictable because of synchronizations).	cmp-<oe_name>
Federation	D4	**Federation of OCI Administrators**	OCI Administrators should be users from the customer Enterprise Identity Management solution and federated with the Default Domain.	root / Default Domain
	D5	**Federation for Workload Identity Stores**	Workload users should be users from the customer Enterprise Identity Management solution and federated with the respective Domain.	cmp-<oe_name> / Workload Domain
Encryption	D6	**Using Customer-Managed Keys**	Customer-managed keys are highly recommended and often mandated by regulations. Separating environment-specific keys in dedicated Vaults may be mandated by the customer. The Vault limits should be considered. Policies can be configured to grant access to specific keys.	cmp-security
Logging and Monitoring	D7	**Integration with External Systems (e.g., SIEM)**	Specific configurations for integrating a customer's SIEM solution must be defined and implemented. Specific configurations for integrating a customer's external solution (e.g., SIEM) must be defined and implemented. If the integration is a general OCI Open LZ level, shared for all OEs, it's recommended to use the shared security compartment (cmp-security) to consolidate all the OCI artifacts for this integration (e.g., Service Connectors, Log Groups, Functions, etc.). If an OE has a requirement to integrate with their own external systems, it's recommended to create a new common security compartment at the OE level (cmp-<oe_name>-common-security) to consolidate all the OCI artifacts for this integration.	cmp-security (for OE Shared integrations) cmp-<oe_name>-common-security (new compartment for OE Dedicated integrations)
	D8	**Events**	A list of defined events (OE specific) must be agreed upon with the customer. Transformation or filtering should be implemented if needed as specified by the customer. These should be placed the in the OE top compartment.	cmp-<oe_name>
	D9	**Logging**	A list of service logs and log groups must be identified. Transformation or filtering should be implemented if needed as specified by the customer. These elements can be placed at the project level, if logs are specific to the projects, business area level, or at shared landing zone level.	cmp-<oe_name>
	D10	**Audit logs**	If needed, Audit logs should be sent to the SIEM. Transformation or filtering should be implemented if needed as specified by the customer.	cmp-security
	D11	**Log archiving**	Archiving of logs may be required by regulations. Using Object Storage with Customer-managed key encryption can be used.	cmp-security
OEs have no responsibility	D12	**Keep using the proposed OE structure**	If organization units exist in the enterprise but they don't operate the environment and resources, we recommend keeping using the proposed model where the OE compartments, users, and groups still exist, but they are all operated by a central team. The advantage of this model is that the	--

over their resources			segregation of resources is still at the OE level (L1), and even operations on different resources can be isolated with sub-team organizations. This path also enables a future-proof design where OEs can eventually inherit responsibility for their resources - with minimum impact.	
	D13	**IT-driven tenancy structure**	If the Landing Zone will not reflect the organization structure, i.e., no OEs are required and D12 is not viable, consider using a tenancy structure driven by environments with projects inside. This model normally points to a centralized operating model by one team, where the business units are just users. Note that moving from D12 to D13 can be a complex tenancy reorganization program and it's not recommended.	--
"Hard" Separation of OEs	D14	**OE-driven tenancy structure**	By default, OEs are placed into a single tenancy using the separation of resources and duties done with OCI policies. There are several considerations that can lead to a **different way of separating OE resources**, such as: • Hosting differently structured organizations can be complex to manage and may also result in too many resources in a single tenancy. • Complete separation of OEs is required by legislation. • Adding new OEs, for example, by acquisition where OCI tenancies already exist. In such situations, it might be easier to **create a tenancy per OE**. These tenancies are grouped as **child tenancies** under a root tenancy which controls the budget of all tenancies. The decision for splitting a single tenancy design into separate child tenancy designs must be done as early as possible and should consider these topics: • No inheritance of configurations and resources from the root tenancy. • No OCI resource migration from one tenancy to another. • Newly created child tenancies follow the same principles as the root tenancy and implement the same resources (CMP.00 - CMP.02, GRP.01 - GRP.07, DGP.01 - DGP.03, POL.01 - POL.07, POL.DGP.01). • Incorporating existing tenancies may result in distinct, inconsistent tenancy architectures. • Additional maintenance effort for the central team on centrally managed resources.	--

Network View

This chapter presents all the network elements of the OCI Open LZ, and it's organized into five sections:

- **Network Structure** presents all the most-significant network components, their relations, and objectives.
- **Network Security** presents the network areas and their related security posture.
- **Network Connectivity** presents how the OCI Open LZ can be connected to on-premises and other cloud providers and describes the most significant network traffic (the network use cases) with a north-south and east-west pattern.
- **DNS** presents the naming resolution use cases and how DNS zones and records are solved to handle domain DNS queries.
- **Design Consideration** presents topics to consider when using, adjusting, or changing this design.

All the resource names in this chapter follow the Oracle Resource Naming Conventions [REF.06] and serve as a guideline.

Network Structure

The OCI Open LZ network structure has the following components:

1. The OCI Open LZ contains a **central network** and **OE-level network** resources, following a **Hub & Spoke** topology, where **an Operating Entity (OE)** is a logical **spoke**.
2. The Hub contains a **central DRG**, a central **VCN,** several Subnets, and capabilities to provide **traffic inspection**, **load-balancing**, and **DNS**.
3. Each **OE** has a set of **dedicated VCNs:** a **common network area** (for possible OE-dedicated inspection, load-balancing, and DNS)**,** a **sandbox area,** and **three environments: development, non-production, and production.**
4. **Each OE Environment VCN** has three **Subnets**, one for each project layer: **App, DB, and Infra**.
5. For each **Subnet,** there is a **common** set of ingress rules handled by **Security Lists**.
6. For each **OCI Open LZ Project,** there will be three **Network Security Groups (NSG)**, one per Project Layer: App, DB, and Infra. With the use of NSGs it's possible to separate project security requirements from the VCNs and Subnets architecture

The following diagram presents an example of the central network components and an OE named "OE01" network components.

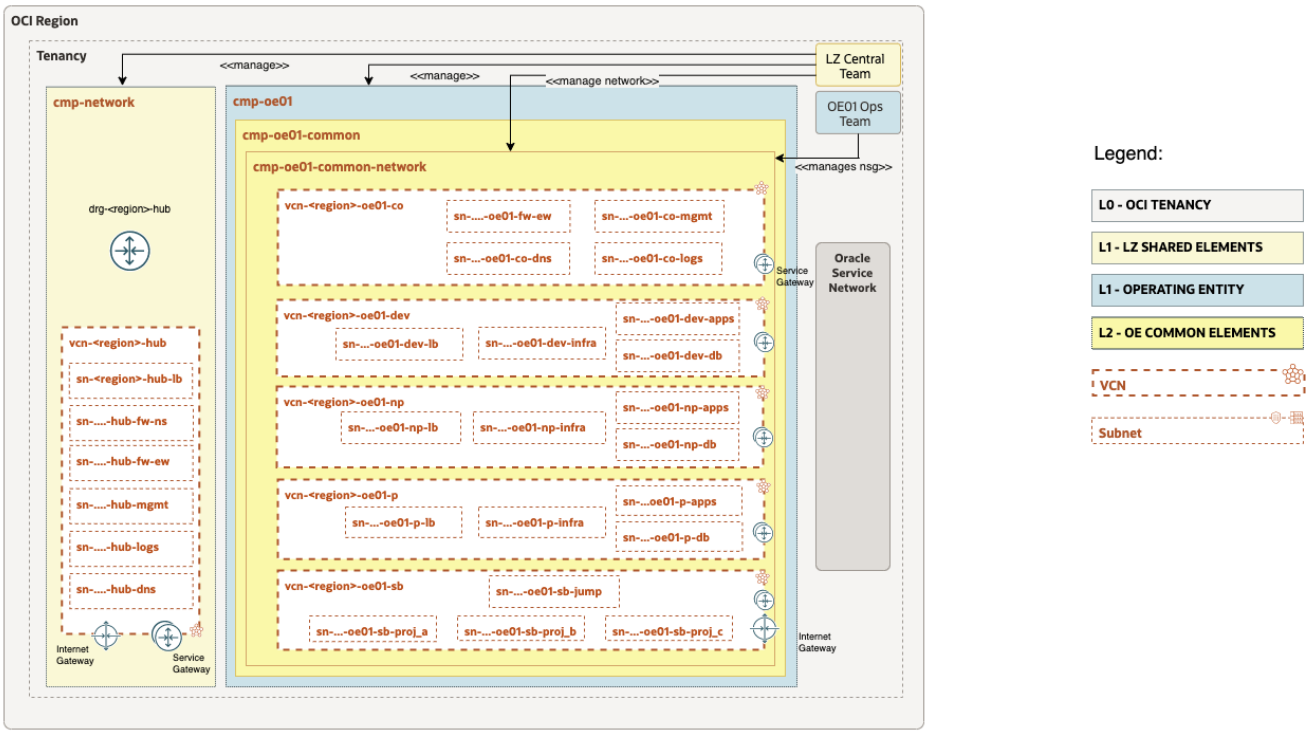


Figure 8 Network View – Network Structure with one OE (OE01).

The diagram below presents how this design is extended to another OE, OE2 in this example. Please note that the following sections do not include this example with a second OE.

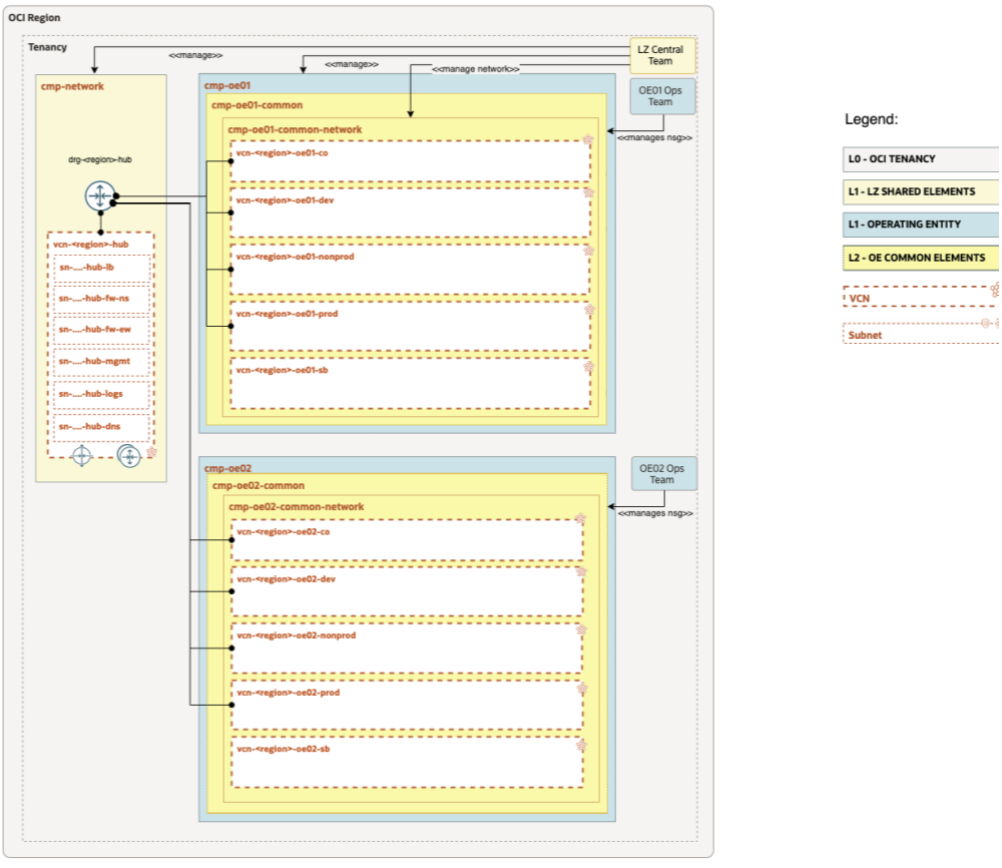


Figure 9 Network View – Network Structure with two OEs.

The network elements presented above are described in the next sections, the Hub, containing the Shared Elements, and the Spokes, containing the Operating Entities resources.

The following abbreviations in the names are suggested due to the maximum length of DNS names of VCN/Subnets (15):

- co = common
- sb = sandbox
- fw = firewall
- np = non-production
- p = production

Hub – Shared Elements

AREA	COMPONEMNT NAME	OCI RESOURCE	DESCRIPTION	TAILOR
Central HUB VCN and DRG.	**vcn-<region>-hub**	VCN	Central HUB VCN for landing zone in the region.	Required
	sn-<region>-hub-lb	Subnet (public)	This element will hold the public Load balancer(s) to handle incoming traffic from the internet.	Recommended
	sn-<region>-hub-fw-ns	Subnet (public)	Network firewall, North-South traffic. This subnet is public to allow incoming traffic from the internet.	Recommended
	sn-<region>-hub-fw-ew	Subnet	Network firewall, East-West traffic.	Recommended
	sn-<region>-hub-mgmt	Subnet	For overall management tasks, such as using Bastion servers.	Recommended
	sn-<region>-hub-logs	Subnet	For SIEM (Security Information and Event Management).	Optional
	sn-<region>-hub-dns	Subnet	For connecting customer DNS servers to OCI. DNS Listener and Forwarders.	Recommended
	drg-<region>-hub	DRG	Main DRG. Connectivity to on-premises or other clouds in the region goes through this DRG (FastConnect or VPN). Also used for remote peering into other regions if needed.	Required
	sgw-<region>-hub	Service Gateway	For access to Oracle Service Network. If need to reach Yum updates for Oracle Linux from Oracle Service Network.	Recommended
	igw-<region>-hub	Internet Gateway	This element should be added if an internet connection is required. To allow inbound or outbound internet traffic.	Optional
	drgatt-<region>-hub-vcn	DRG attachment	Enable the connection between VCN and DRG so network traffic can flow.	Required

Spokes – Operating Entities

AREA	COMPONEMNT NAME	OCI RESOURCE	DESCRIPTION	TAILOR
OE COMMON - Common network resources for each OE	**vcn-<region>-oe01-co**	VCN	VCN to host common resources for an Operating Entity.	Required
	sn-<region>-oe01-fw-ew	Subnet	Subnet to host Network firewall, East-West traffic	Optional
	sn-<region>-oe01-co-mgmt	Subnet	Subnet to host management resources for an Operating Entity.	Required
	sn-<region>-oe01-co-logs	Subnet	For SIEM (Security Information and Event Management) and log systems for an Operating Entity.	Recommended
	sn-<region>-oe01-co-dns	Subnet	For connecting on-premises DNS servers to OCI. DNS Listener and Forwarders.	Recommended
	sgw-<region>-oe01-co	Service Gateway	For access to Oracle Service Network. If need to reach Yum updates for Oracle Linux from Oracle Service Network.	Recommended
	drgatt-<region>-oe01-co-vcn	DRG attachment	Enables the connection between VCN and DRG so network traffic can flow.	Required
OE SANDBOX - Network resources for externally accessed test env for projects/demo that is not allowed to touch the internal network. No DRG is attached to this VCN, only an Internet gateway.	**vcn-<region>-oe01-sb**	VCN	CN for sandbox environments (PoC/test/dev) for OE01.	Optional
	sn-<region>-oe01-sb-jump	Subnet	Public Subnet for Bastion servers into Project private subnets.	Optional
	sn-<region>-oe01-sb-<proj_a>	Subnet	Private subnet for Project A environment for OE01.	Optional
	sn-<region>-oe01-sb-<proj_b>	Subnet	Private subnet for Project B environment for OE01.	Optional
	sn-<region>-oe01-sb-<proj_c>	Subnet	Private subnet for Project C environment for OE01.	Optional
	ig-<region>-oe01-sb	Service Gateway	To allow incoming traffic through Bastion servers in a public subnet, to reach private sandbox project subnets in OE01.	Optional
	sgw-<region>-oe01-sb	DRG attachment	For access to Oracle Service Network. Oracle Linux Yum updates. Object storage. Backup of DB.	Optional
OE DEVELOPMENT ENVIRONMENT - Network resources for Development environments	**vcn-<region>-oe01-dev**	VCN	VCN for development environments for OE01.	Optional
	sn-<region>-oe01-dev-lb	Subnet	Subnet for load balancers on the development environments for OE01.	Optional
	sn-<region>-oe01-dev-infra	Subnet	Subnet for infrastructure resources on the development environments for OE01.	Optional
	sn-<region>-oe01-dev-app	Subnet	Subnet for application resources on the development environments for OE01.	Optional
	sn-<region>-oe01-dev-db	Subnet	Subnet for database resources on the development environments for OE01.	Optional
	sgw-<region>-oe01-dev	Service Gateway	For access to Oracle Service Network. Oracle Linux Yum updates. Object storage. Backup of DB	Optional
	drgatt-<region>-oe01-dev-vcn	DRG attachment	Enables the connection between VCN and DRG so network traffic can flow.	Optional

OE NON-PRODUCTION ENVIRONMENT - Network resources for non-production environments.	**vcn-<region>-oe01-np**	VCN	VCN for development environments for OE01.	Required
	sn-<region>-oe01-np-lb	Subnet	Subnet for load balancers on the non-production environments for OE01.	Required
	sn-<region>-oe01-np-infra	Subnet	Subnet for infrastructure resources on the non-production environments for OE01.	Required
	sn-<region>-oe01-np-app	Subnet	Subnet for application resources on the non-production environments for OE01.	Required
	sn-<region>-oe01-np-db	Subnet	Subnet for database resources on the non-production environments for OE01.	Required
	sgw-<region>-oe01-np	Service Gateway	For access to Oracle Service Network. Oracle Linux Yum updates. Object storage. Backup of DB	Required
	drgatt-<region>-oe01-np-vcn	DRG attachment	Enables the connection between VCN and DRG so network traffic can flow.	Required
OE PRODUCTION ENVIRONMENT - Network resources for Production environments.	**vcn-<region>-oe01-p**	VCN	VCN for development environments for OE01.	Required
	sn-<region>-oe01-p-lb	Subnet	Subnet for load balancers on the production environments for OE01.	Required
	sn-<region>-oe01-p-infra	Subnet	Subnet for infrastructure resources on the production environments for OE01.	Required
	sn-<region>-oe01-p-app	Subnet	Subnet for application resources on the production environments for OE01.	Required
	sn-<region>-oe01-p-db	Subnet	Subnet for database resources on the production environments for OE01.	Required
	sgw-<region>-oe01-p	Service Gateway	For access to Oracle Service Network. Oracle Linux Yum updates. Object storage. Backup of DB	Required
	drgatt-<region>-oe01-p-vcn	DRG attachment	Enables the connection between VCN and DRG so network traffic can flow.	Required

Network Security

This section presents the OCI Open LZ network areas and their security posture. The area dedicated to projects, that will contain workloads, will be described in more detail for a clear understanding of how workloads run on the landing zone, which elements they share, which elements secure them. To complement this last view, refer to the Most-Significant Network Traffic (North-South / East-West) to understand how data flows in the OCI Open LZ with these artefacts.

Network Areas

The OCI Open LZ has **four network areas** which are presented and described in the next diagram and table, respectively.

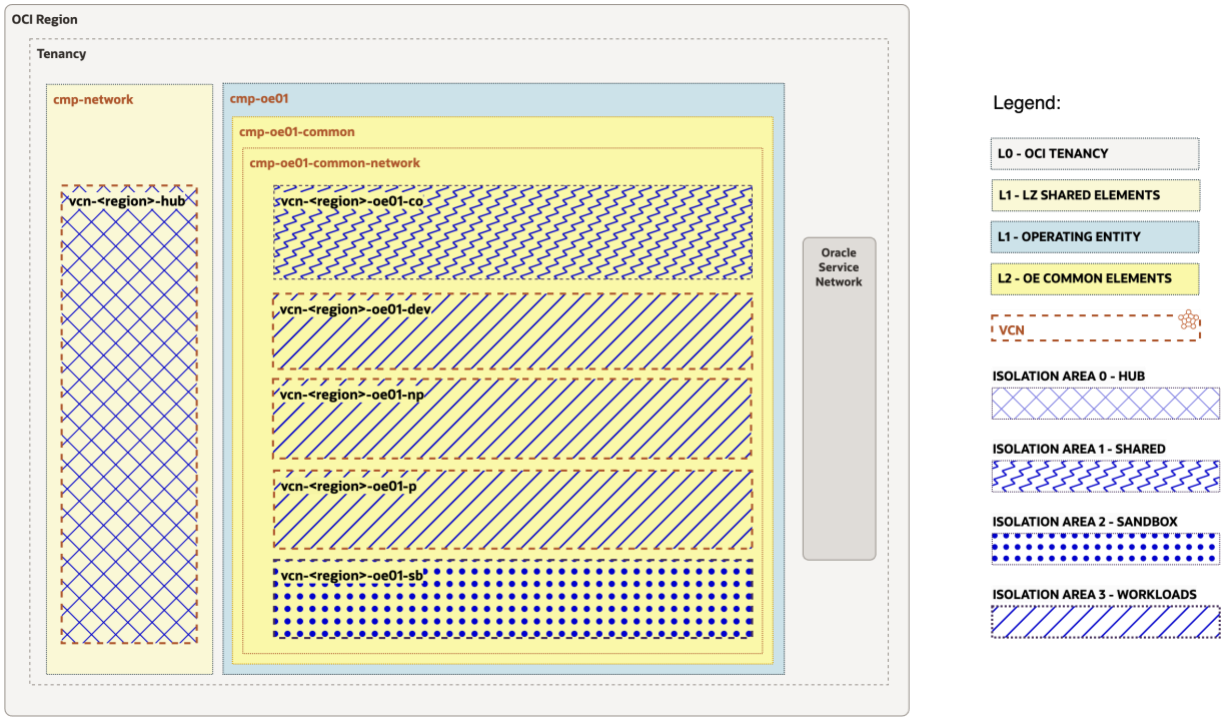


Figure 10 Network View – Network Areas.

AREA	USED FOR	OBJECTIVE
0 – Hub	Central communication and connectivity place for an OCI Open LZ region.	• Central routing, log, management, and DNS area in the region. • Central firewall/inspection of traffic to-from on-premises and between OEs. • Inspection of traffic to/from the internet. • A place for DNS listener/forwarder
1 – OE Shared Elements	OE common network resources and connectivity.	• An area for OE logs, Bastion service, DNS Listener/Forwarders, and firewall if needed. • Control traffic between OE Projects if needed through the firewall. • Bastion Service: In case of use, there would be one instance per OE Environment - Layer target (VCN - Subnet).
2 – Projects / Workloads	OE Workloads, on Development, Non-production, and Production Environments.	• Provide network control and isolation of OE project resources. • Service Gateway is attached for Object storage, backup, or Oracle Linux yum server purposes
3 – Sandbox	OE Workloads, on Sandbox / PoC area for internet-exposed environments.	• Isolated area for PoC/sandbox projects. • No contact to internal OCI/on-premises network allowed. Only from the internet.

Network Security Posture

The table below presents the network security posture for each area. The security posture is divided into two categories: connectivity and security elements used. For a visual classification, the **light blue** is used to reflect more network "freedom" while the **purple** reflects more control capabilities, meaning a stronger network security.

SECURITY POSTURE AREAS	CONNECTIVITY				SECURITY		
	DRG ATTACHMENT	INTERNET GATEWAY	NAT GATEWAY	SERVICE GATEWAY	NSG	SECURITY LIST	NETWORK FIREWALL
0 – Hub	Yes	Yes (Optional)	No (Optional)	Yes	Yes	Yes	Yes
1 – OE Shared Elements	Yes	No	No	Yes	Yes	Yes	Yes (Optional)
2 – Projects / Workloads	Yes	No	No	Yes	Yes	Yes	No
3 – Sandbox	No	Yes	No	Yes	No	Yes	No

Projects/Workloads Isolation (Area 2)

The following diagram presents an example of the **network structure** for the **OE01** project **P1** on **production,** reflecting the principles above, where instances reside, in compartments but also VCNs and Subnets.
For the non-production environment, the same rules would apply but the "P1" resources would be associated with the non-production OE VCN and compartment.

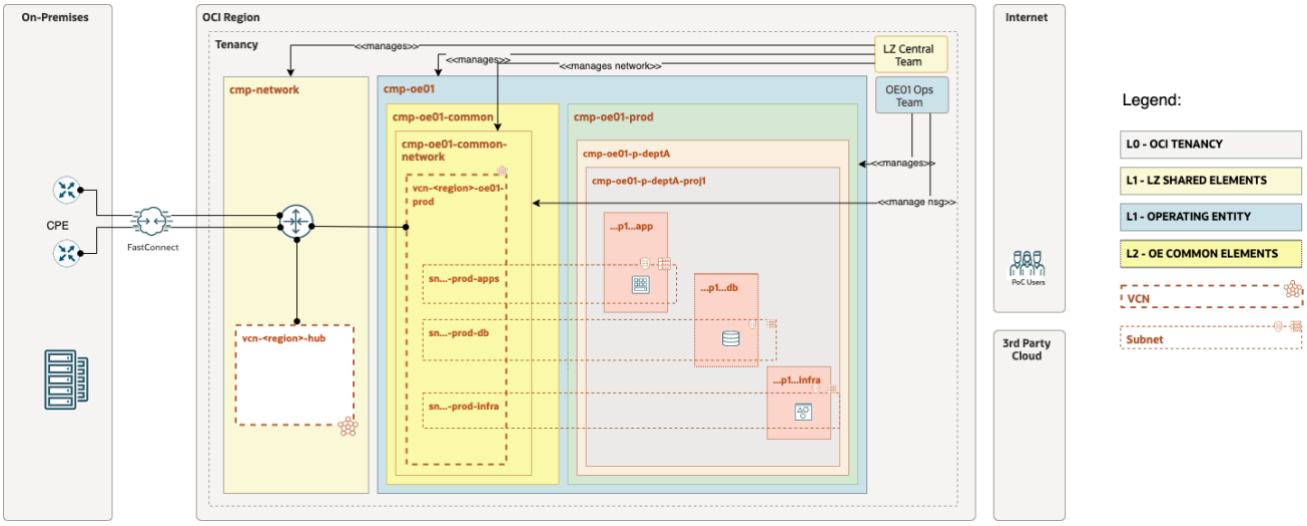


Figure 11 Network View – Project Isolation (VCNs and Subnets).

The diagram below with project **P1** as an example presents the **network isolation**, implemented at the project level with **dedicated NSG per project layer**.

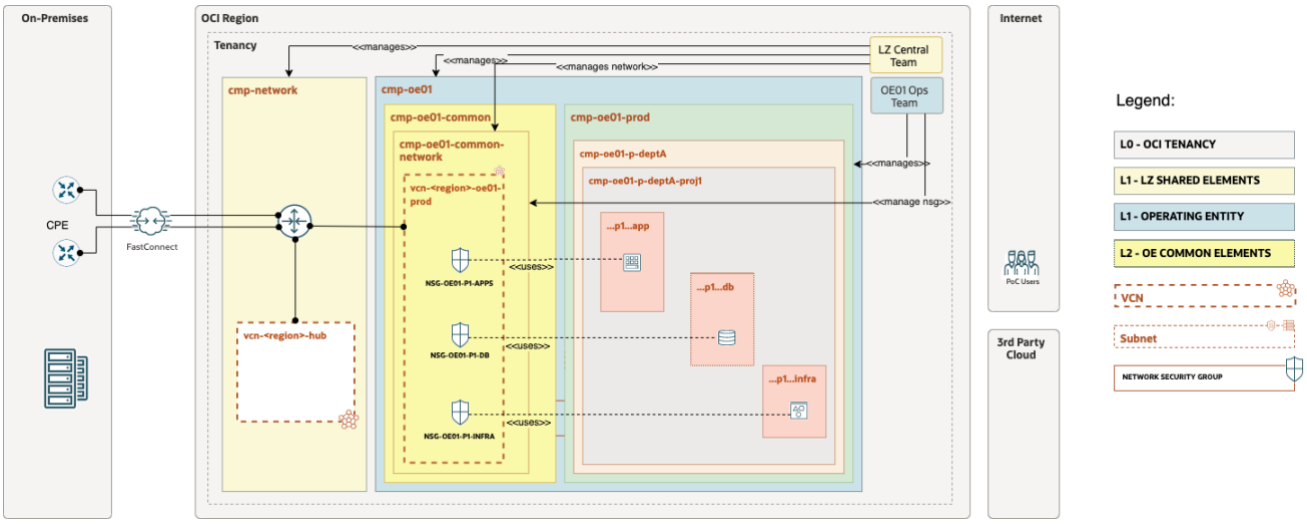


Figure 12 Network View – Project Isolation (NSGs).

Note that OE VCNs (production and non-production) and Subnets are shared by projects at the OE level, as depicted in the following diagram for projects **P1** and **P2 application layer,** sharing the **OE application Subnet.**

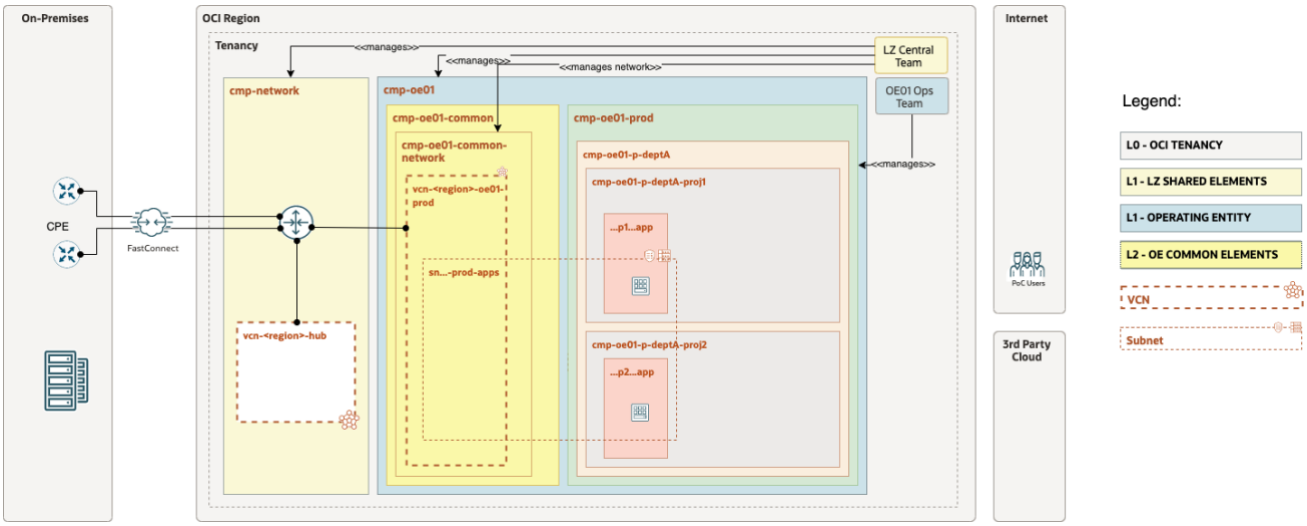


Figure 13 Network View – Project Isolation and Shared Resources.

Network Connectivity

OCI Connectivity to Other CSP / On-prem

This section presents how OCI connects with on-premises and/or other cloud service providers. The following diagram represents an example.

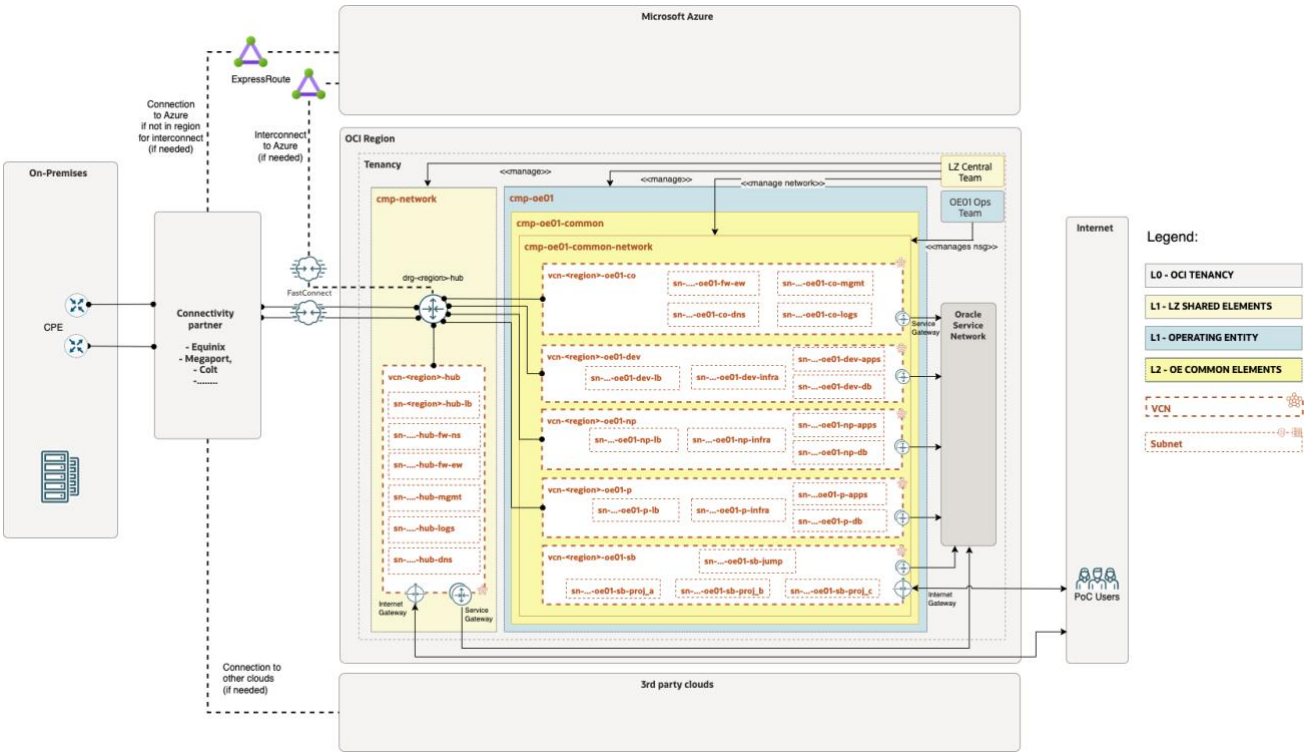


Figure 14 Network View – Connectivity to On-premises/CSP.

Microsoft Azure

Oracle has a partnership with Microsoft that makes it easy without and 3rd party connectivity provider to connect Oracle cloud with Microsoft Azure if using regions where the interconnect is present.

For more details refer to Access to Microsoft Azure.

Other CSPs and On-premises Data Centers

Most of the FastConnect partners can be used to connect to other CSPs. The same strategy applies to connecting customers' on-premises data centers to OCI. Refer to the list of FastConnect partners per region and location for more details.

Most-Significant Network Traffic (North-South / East-West)

The objective of this section is to simplify the understanding of OCI Open LZ network connectivity while explaining the OCI network elements' relations and dependencies. To achieve this:

- The **most-significant OCI Open LZ network traffic scenarios are identified** and presented with a description of the participating OCI network elements, for each scenario.
- Each **scenario,** or **network use case, is classified** as **North-South (NS)** for data flow from or to resources outside OCI, or **East-West** (EW) for traffic between resources inside OCI.
- The examples follow an inbound or outbound flow from a **project application layer**.

Note that i) these descriptions aim to simplify the interpretation of the OCI Open LZ, and they do not intend to describe or specify OCI's detailed/actual internal flows, and ii) these are stateful communications examples so that the reply message would follow the inverse flow.

NS.1 – North-South With On-Premises/CSP (Inbound)

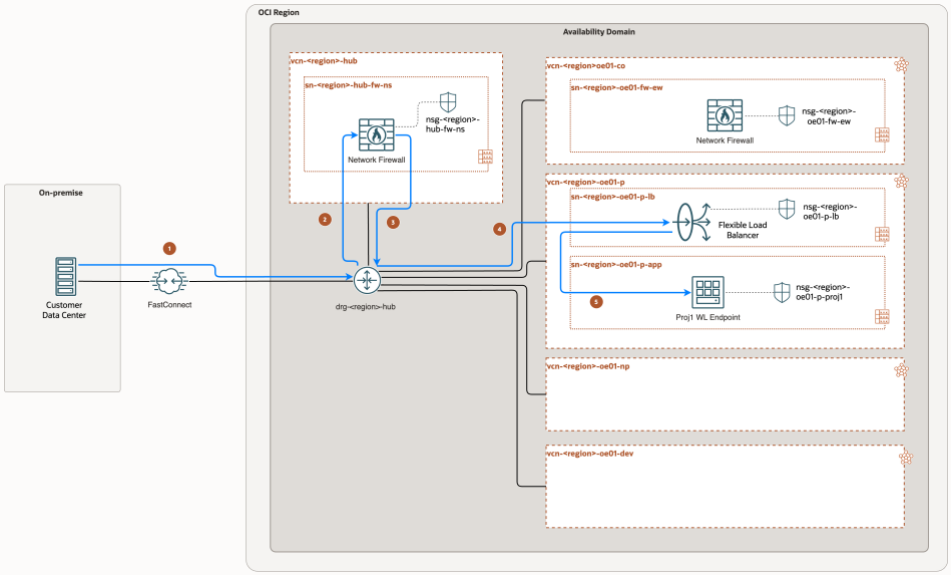
USE CASE (FROM -> TO)	**(On-Prem/CSP → OE01 Proj1)** **OCI Inbound On-prem/CSP Traffic to an OE Project, Application layer.**
SEQUENCE FLOW – HIGH LEVEL	**(Client)** → **(Fastconnect)** → (Hub **DRG)** → (**Hub VCN**) → (**North-South Network FW**) → (Hub **DRG**) → (OE01 Env. VCN / **LB Subnet**) → (OE01 Private **Load Balancer)** → (OE01 Env. VCN / **App Subnet**) → (OE01 Proj1 / App **WL Endpoint).**
SEQUENCE FLOW - DETAILED (with, NSGs, Route Tables, Ingress/Egress in blue)	**(Client)** → **(Fastconnect)** → (Hub **DRG /** FC Attch RT**)** → (**Hub VCN** RT.Ingress) → (NS-**NFW NSG**.Ingress) → (**NS - Network FW**) → (NS-**NFW NSG**.Egress) → (**FW-NS Subnet** RT.Egress) → (Hub **DRG** / Hub VCN Attch RT **)** → (OE01 Env. VCN / LB Subnet / **LB NSG**.Ingress) → (OE01 Private **Load Balancer)** → (OE01 Env. VCN / LB Subnet / **LB NSG.**Egress) → (OE01 Env. VCN / App Subnet / Proj1 **App NSG.**Ingress) → (OE01 Proj1 **WL** Endpoint).
SEQUENCE DIAGRAM	

Figure 15 Network View – North-South with On-Premises/CSP (Inbound).

NS.2 – North-South With On-Premises/CSP (Outbound)

USE CASE (FROM -> TO)	**(OE01 Proj1 → On-Prem/CSP)** **OCI Outbound Traffic from an OE Project to On-Premises/CSP**
SEQUENCE FLOW – HIGH LEVEL	(OE01 Proj1 **WL** Endpoint) → (Hub **DRG)** → (Hub VCN / **FW-NS Subnet**) → (**North-South Network FW**) → (Hub **DRG**) → **(Fastconnect)** → **(Client)**
SEQUENCE FLOW - DETAILED (with, NSGs, Route Tables, Ingress/Egress in blue)	(OE01 Proj1 **WL** Endpoint) → (OE01 Env. VCN/Proj1 **App NSG.**Egress) → (OE01 Proj1 Env. VCN/**Subnet** RT.Egress) → (Hub **DRG /** OE01 VCN Attach RT**)** → (Hub **DRG)** → (Hub **VCN** / RT.Ingress) → (NS-**NFW**

	NSG.Ingress) → (**NS-NFW**) → (NS-NFW NSG.Egress) → (Hub VCN / **FW-NS Subnet** RT.Egress) → (Hub **DRG /** Hub VCN Attach.RT) → **(Fastconnect) → (Client)**
SEQUENCE DIAGRAM	Refer to NS.1 reference diagram.

NS.3 – North-South With Internet (Inbound)

USE CASE (FROM -> TO)	**(Internet → OE01 Proj1)** **OCI Inbound Internet Traffic to an OE Project, Application layer.** This scenario is required if there is a need for internet traffic to reach projects in one OE environment. Network layer 7 will be used for communication.
SEQUENCE FLOW – HIGH LEVEL	**(Client) → (Internet) → (Internet Gateway** / Hub VCN) → (Hub VCN / **LB Public Subnet**) → (Public **LB**) → (Hub VCN / **FW-NS Subnet**) → (**North-South Network FW**) → (Hub **DRG**) → (OE01 Env. VCN / App Subnet) → (OE01 Proj1 / App **WL Endpoint).**
SEQUENCE FLOW - DETAILED (with, NSGs, Route Tables, Ingress/Egress in blue)	**(Client) → (Internet) → (IG** / Hub VCN / IG RT.Ingress) → (Hub VCN) → (Hub VCN / **LB Public Subnet** LB-NSG.Ingress) → (Public **LB**) → (Hub VCN / **LB Public Subnet** LB-NSG.Egress) → (Hub VCN / **LB Public Subnet** RT.Egress) → (Hub VCN / **FW-NS Subnet** RT.Ingress) → (NS-NFW NSG.Ingress) → (**NS-NFW**) → (NS-NFW NSG.Egress) → (Hub VCN / **FW-NS Subnet** RT.Egress) → (Hub **DRG /** Hub VCN Attach.RT**)** → (Hub **DRG)** → (OE01 Env. VCN / App Subnet / Proj1 App NSG.Ingress) → (OE01 Proj1 **WL** Endpoint).
SEQUENCE DIAGRAM	Refer to NS.1 reference diagram.

NS.4 – North-South With Internet (Outbound)

USE CASE (FROM -> TO)	**(OE01 Proj1 → Internet)** **OCI Outbound Traffic from an OE Environment to the Internet** Note that it's required to define when and how traffic to and from the internet is allowed. In this pattern, there is an Internet Gateway attached to Hub VCN in the diagram, but it's declared as optional. This Internet Gateway can be used for both incoming and outgoing traffic. Note that if there is a need to only have internet traffic where the initiation is from within OCI, then a NAT gateway is the best option (traffic is not allowed to be initiated from the internet with NAT Gateways).
DEFAULT SEQUENCE FLOW – HIGH LEVEL	Identical to traffic flow NS.2. All outbound internet traffic goes to on-premises and then redirected to the internet.
ALTERNATIVE FLOW – HIGH LEVEL	This flow uses Internet Gateway in Hub VCN to allow Internet traffic initiated from within OCI. Note that this option can allow internet-initiated traffic to flow into VCNs if ingress rules are added. (OE01 Proj1 **WL** Endpoint) → (Hub **DRG)** → (Hub VCN / **FW-NS Subnet**) → (**North-South Network FW**) → (Hub VCN / **IG**) → **(Internet)**
SEQUENCE FLOW - DETAILED (with, NSGs, Route Tables, Ingress/Egress in blue)	(OE01 Proj1 **WL** Endpoint) → (OE01 Env. VCN/Proj1 App NSG.Egress) → (OE01 Proj1 Env. VCN/**Subnet** RT.Egress) → (Hub **DRG /** OE01 VCN Attach RT**)** → (Hub **DRG)** → (Hub VCN / **FW-NS Subnet** RT.Ingress) → (NS-NFW NSG.Ingress) → (**NS-NFW**) → (NS-NFW NSG.Egress) → (Hub VCN / **FW-NS Subnet** RT.Egress) → **(** Hub VCN **IG** RT.Egress**)** → **(Internet)**
SEQUENCE DIAGRAM	Refer to NS.1 reference diagram.

NS.5 – North-South With Sandbox (Inbound)

USE CASE (FROM -> TO)	**(PoC Users → OE01 Sandbox)** **OCI Inbound Traffic to the Sandbox environment.** The PoC/sandbox environment does not have any connections to the internal network. Access is only allowed from the Internet. Has a public subnet for use with Bastion servers. The intention is to have a completely isolated environment from the internal network and have an internet-facing sandbox/PoC project place.
SEQUENCE FLOW – HIGH LEVEL	**(Client) → (Internet) → (IG** / OE01 Sandbox VCN) → (OE01 Sandbox VCN/**Jump Server Subnet**) → (**Jump Server**) → (OE01 Sandbox VCN/**Project Subnet**) → (OE01 Sandbox Project **WL** Endpoint).
SEQUENCE FLOW - DETAILED (with, NSGs, Route Tables, Ingress/Egress in blue)	**(Client) → (Internet) → (IG** / OE01 Sandbox VCN / IG) → (OE01 Sandbox VCN/**Jump Server Subnet** NSG.Ingress) → (OE01 Sandbox VCN/**Jump Server Subnet** NSG.Egress) (OE01 Sandbox VCN/**Project Subnet** RT.Ingress) → (OE01 Sandbox Project **WL** Endpoint).
SEQUENCE DIAGRAM	Refer to NS.1 reference diagram.

EW.1 – East-West, with two OEs (Inter-OE)

USE CASE (FROM -> TO)	**(OE01 Proj 1 → OE02 Proj 3)** **Traffic from one Project to another Project, on different OEs.** This scenario is required if there is a need for projects in one OE to communicate with projects in another OE and will use network layer 7 for communications with NSG allowing this traffic. The Proj1 Workload will only be aware of the Proj3 Load Balancer Virtual Hostname, which will act as a rule to determine the final endpoint by the load balancer. This flow will use the Hub DRG and will be subject to traffic inspection by the Hub EW Firewall. Note that if the OE02 would sit in another OCI Region, there would be an RPC in the flow, between the two regional DRGs.
SEQUENCE FLOW – HIGH LEVEL	(OE01 Proj1 **WL** Endpoint) → (Hub **DRG)** → (Hub VCN / **FW-EW Subnet**) → (**East-West Network FW**) → (Hub **DRG)** → (OE02 Env. VCN / **LB Subnet**) → (OE02 Private **LB)** → (OE02 Env. VCN / **App Subnet**) → (OE02 Proj3 **WL** Endpoint).
SEQUENCE FLOW - DETAILED (with, NSGs, Route Tables, Ingress/Egress in blue)	(OE01 Proj1 **WL** Endpoint) → (OE01 Env. VCN/Proj1 **App NSG.**Egress) → (OE01 Env. VCN/**App Subnet** RT.Egress) → **(OE01 Env. VCN** Attch RT) → (Hub **DRG)** → (Hub VCN / **FW-EW Subnet** RT.Ingress) → (**EW-NFW NSG.**Ingress) → (**EW-NFW**) → (**EW-NFW NSG.**Egress) → (Hub VCN / **FW-EW Subnet** RT.Egress) → (Hub **DRG** / Hub VCN Attach RT**)** → (OE02 Env. VCN / **LB Subnet / LB NSG.**Ingress) → (OE02 Private **LB)** → (OE02 Env. VCN / **LB Subnet / LB NSG.**Egress) → (OE02 Env. VCN / App Subnet / Proj3 **App NSG.I**ngress) → (OE02 Proj3 **WL** Endpoint).

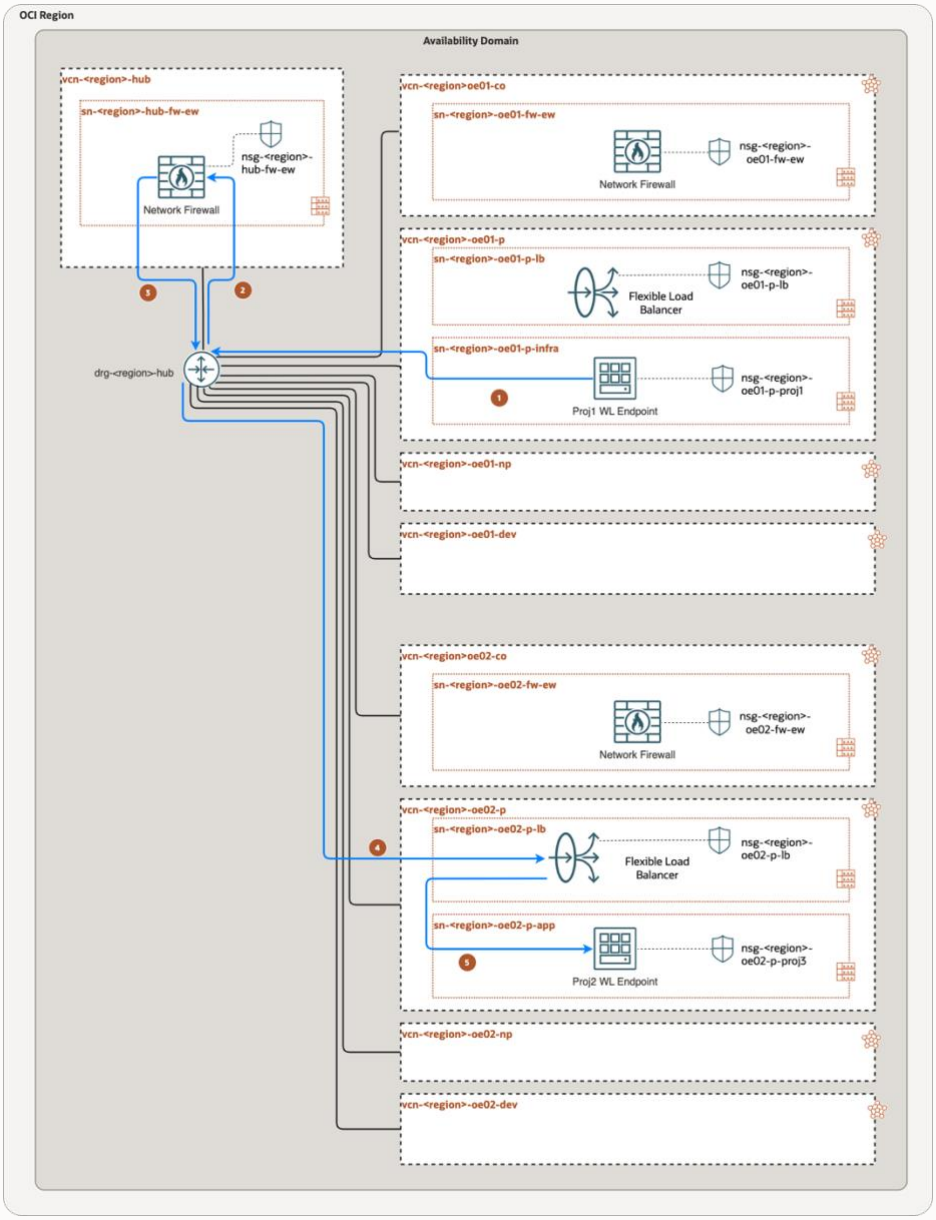
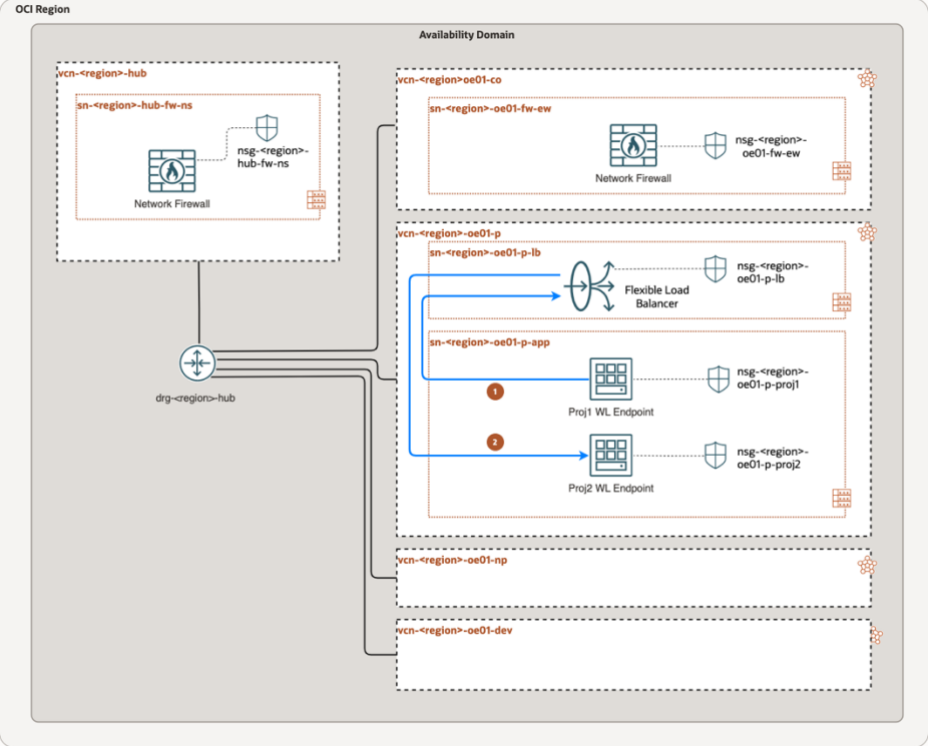
SEQUENCE DIAGRAM	

Figure 16 Network View – East-West, with two OEs (Inter-OE).

EW.2 – East-West in one OE (Intra-OE)

USE CASE (FROM -> TO)	**(OE01 Proj 1 → OE01 Proj 2)** **INTRA OE Traffic: from one Project to another Project, inside an OE** This scenario is required if there is a need for projects in the same OE to communicate with each other in the same or different environments but in the same VCN. The use of a firewall to inspect traffic is a design decision. This scenario will use a network layer 7 load balancer for communication between endpoints. This scenario forwards traffic to an endpoint by forwarding the request to LB instead to the application directly. The LB will forward traffic to the right backends.
SEQUENCE FLOW – HIGH LEVEL	(OE01 App Subnet / Proj1 **WL** Endpoint) → (OE01 Env. VCN / **LB Subnet**) → (OE1 Private **LB)** → (OE01 App Subnet) → (OE01 Proj2 **WL** Endpoint).

SEQUENCE FLOW - DETAILED (with, NSGs, Route Tables, Ingress/Egress in blue)	(OE01 Proj1 **WL** Endpoint) → (OE01 Env. VCN/Proj1 **App NSG.**Egress) → (OE01 Env. VCN / **LB Subnet** / **LB NSG.**Ingress) → (OE01 Private **LB)** → (OE01 Env. VCN / LB Subnet / **LB NSG** Egress) → (OE01 App Subnet / Proj2 App **NSG.**Ingress) → (OE01 Proj2 **WL** Endpoint).
SEQUENCE DIAGRAM	 Figure 17 Network View – East-West in one OE (Intra-OE).
ALTERNATIVE SEQUENCE FLOW A) – HIGH LEVEL	**Alternative Flow C - Shared Hub EW-FW**: Identical to scenario EW.1.
ALTERNATIVE SEQUENCE FLOW B) – HIGH LEVEL	**Alternative Flow B - OE Dedicated EW-FW**: (OE01 Proj1 **WL** Endpoint) → (OE01 Common VCN / **FW-EW Subnet**) → (**East-West Network FW**) → (OE01 Env. VCN / **LB Subnet**) → (OE01 Private **LB)** → (OE01 Env. VCN / App Subnet) → (OE01 Proj2 **WL** Endpoint).
ALTERNATIVE SEQUENCE FLOW B) – DETAILED	**Alternative Flow B - OE Dedicated EW-FW**: (OE01 Proj1 **WL** Endpoint) → (DSN Resolver: Hostname to IP) → (OE01 Env. VCN / App Subnet / Proj1 **App NSG.**Egress) → (OE01 Env. **VCN / App Subnet** RT.Egress) → (OE01 Common VCN / **FW-EW Subnet** RT.Ingress) → (**EW-NFW NSG.**Ingress) → (**EW-NFW**) → (**EW-NFW NSG.**Egress) → (OE01 Env. VCN / LB Subnet / **LB NSG.**Ingress) → (OE01 Private **LB)** → (OE01 Env. VCN / LB Subnet / **LB NSG**.Egress) → (OE01 Env. VCN / App Subnet / Proj 2 App **NSG.**Ingress) → (OE01 Proj2 **WL** Endpoint).

EW.3 – East-West with OSN

USE CASE (FROM -> TO)	**(OE01 Proj1 → OSN)** **Traffic to Oracle Service Network** This traffic is used when doing backup of databases to Object storage and when doing OS updates of Oracle Linux. It can also be used for other standard Object storage usage and use of other services in the Oracle Service Network.
SEQUENCE FLOW – HIGH LEVEL	(OE01 Proj1 **WL** Endpoint) → (OE01 Env. VCN/**Service Gateway**) → (**OSN)**

SEQUENCE FLOW - DETAILED (with, NSGs, Route Tables, Ingress/Egress in blue)	(OE01 Proj1 **WL** Endpoint) → (OE01 Env. VCN/Proj1 **App NSG.**Egress) → (OE01 Env. VCN/Proj1 RT.Egress) → (OE01 Env. VCN/**Service Gateway**) → (**OSN**)
SEQUENCE DIAGRAM	--

Domain Name System (DNS)

This section presents the private DNS for the OCI Open LZ. The Public DNS setup/configuration is not part of this landing zone but can be added after the landing zone has been deployed. For more information on public DNS refer to the public documentation.

If an OE needs to connect their DNS system - either on-premises or on other clouds - the name resolutions can be achieved on OCI resources from their clients, from the customer networks, or from within OCI, using the guidelines described in the next sections. In this description, we explain the setup for one OE, nevertheless, t**he same setup is recommended for the parent company that manages the Hub VCN**. The Sandbox environment will **not** be part of the private DNS setup since it's completely separated from the internal network.

Name Resolution within an OE

The name resolution within an OE follows these criteria:

1. **Private views** are a collection of **Private DNS zones**. When creating a VCN, a private view is automatically created, and subnets that are created within that VCN become private DNS zones.
2. **Resources created in a VCN**, under a specific subnet, will automatically get an **FQDN** (Fully Qualified Domain Name) populated in the **private DNS zone** for that subnet.
3. **Name resolution within a VCN** is then automatically set up for the standard FQDN (Fully Qualified Domain Names).
4. To set up **name resolution between different VCNs** but within a region, private views need to be associated. If name resolution between VCNs is not required, leave the specific VCN out of this association.
5. Alternative to association between private views is to use forwarding rules.
6. We propose a combination. Forwarding rules from all spokes to common VCN (forward all requests). Common VCN is associated with all spokes so they can do lookups for all spokes and respond.
   This requires forwarding rules only in common VCN if need to add or change any rules.

The following table presents the OCI Open LZ private view associations example:

OE-VCN	ASSOCIATED WITH
vcn-<region>-<OE>-co	**vcn-<region>-<OE>-dev** **vcn-<region>-<OE>-np** **vcn-<region>-<OE>-p**

Location of DNS Resources for an OE

The location of DNS resources for an OE follows these guidelines:

1. **The common VCN** will be used as a shared area for **DNS resources** that are used f**or an OE** (vcn-<region>-<OE>-co).
2. Within that common VCN, there is a **DNS subnet** (sn-<region>-<OE>-dns) where the **listener and forwarder DNS endpoints** will be created.

3. If needed, **private DNS zones** are created under a manually added **private view** in the region. Then associated this private view with the one in Common VCN.
4. **Forwarding rules** are created in all spokes to forward all DNS lookups to one place in the region, to common VCN resolver.
5. **Forwarding rules** are created in common VCN resolver to forward DNS request to another DNS server if needed.

The diagram below shows DNS resources in one region :

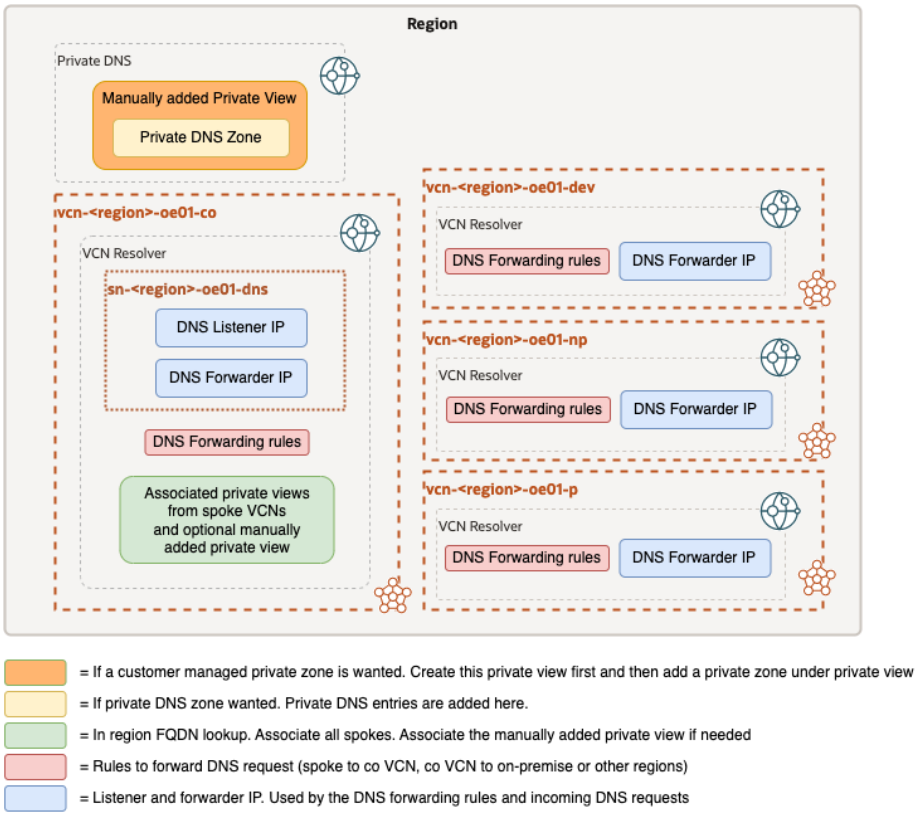


Figure 18 Network View – DNS Resources in one region.

DNS To/From Outside an OCI Region

If an on-premises server needs to resolve an OCI resource for a specific OE, it forwards the request from the on-premises DNS server to the OCI **listener IP** in dns subnet in OE common VCN.

The same setup is done between two OCI regions. Forward the request from the **forwarder IP** in the source OCI region to the remote region **listener IP**. The DNS forwarding rules in each region need to be set up to trigger forwarding.

The On-premises DNS servers or DNS in other clouds, need to accept DNS requests coming from the **forwarder IP** in the dns subnet, if OCI resources should be able to resolve resources outside of their OCI OE environment.

The following diagram shows the DNS traffic flows if name resolution from a domain outside or between OCI regions is required.

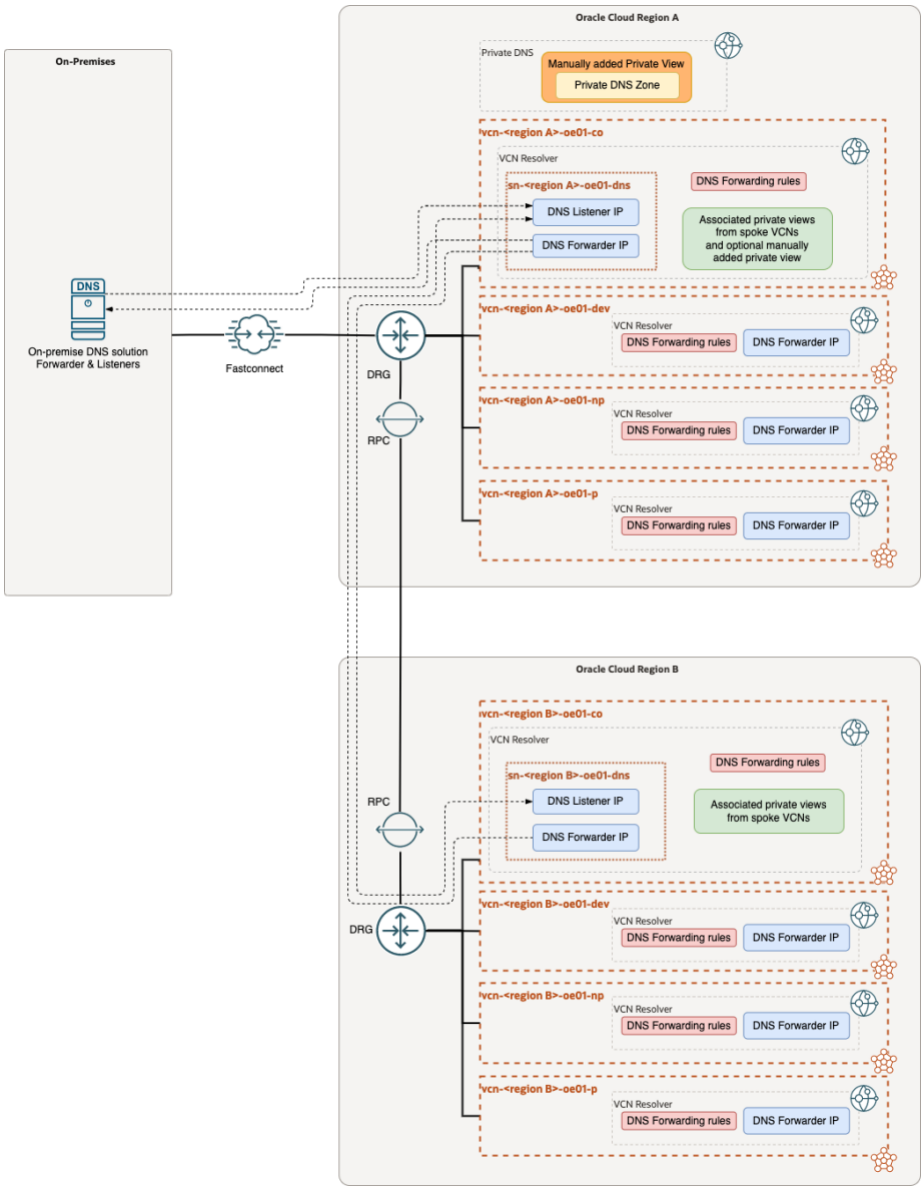


Figure 19 Network View – DNS traffic with name resolution from a domain outside or between OCI regions.

DNS Forwarding Rules

When resources within OCI need to resolve private DNS names outside of their OCI/OE environment or in another OCI region, it needs to be forwarded to another DNS server for resolution.

The rules for OCI/OE env should be configured in the common VCN DNS resolver (vcn-<region>-<OE>-co).

To add a rule, a **listener IP** (Destination IP address) for an on-premises DNS server (or OCI remote region) is needed, with one or more domain names to trigger the rule and a **forwarder IP**.
Below is an example of such a rule.

Design Considerations

The previous sections presented an OCI cloud-native network design. There are several alternatives to this model, based on possible requirements. Consider the following topics when adjusting the OCI Open LZ.

TOPIC	ID	DECISION	GUIDELINES
Third-Party Firewalls	D1	Use Third-Party Firewalls	Use third-party firewalls with the same network design used above, replacing OCI NFW with these elements. Note that each third-party firewall might require additional resources for its operations, for example: • **Subnets:** Add the required new subnets at the same as the firewall main subnet, following section Network Structure. • **Virtual Machines:** At the Hub level, the required compute virtual machines should be placed on CMP.02 "**cmp-security**". At OE-level they should be placed on CMP.02 **"cmp-<oe_name>-common-network".**
Segregation of Duties for NS and EW	D2	Create two Hub compartments (NS and EW)	If the teams responsible for North-South traffic are not the same as the ones that manage East-West traffic, it's recommended to create two hub compartments to place all the related resources (VCNs, Subnets, RTs, SLs, Firewalls, DRG Attachments, etc.) on each and associated them to different groups and policies.

Operations View

This chapter presents what cloud-native operations look like with the OCI Open LZ pattern and it's organized into four sections:

- **Cloud-Native Operations** presents an overview of what modern cloud operations look like. These concepts are used throughout this chapter and are key to building the target OCI Open LZ operating model and Runtime View.
- **Provisioning and Change** identifies the key OCI Open LZ operations building blocks, i.e., the operations scenarios (what), the responsible teams to execute them (who), and how to automate these operations (how).
- **Operating Model** builds on top he the previous section and presents the target operating model for the OCI Open LZ using a set of git repositories - that act as the single source of truth for operations. This section provides the operational controls required to operate at scale, as simple automation of operations scenarios described in Provisioning and Change section is not enough.
- **Design Considerations** presents the design rationale to use when tailoring or extending this pattern.

Cloud-Native Operations

Cloud-native operations on cloud resources should be very near the development practices, and therefore the target operating model, for **dev** or **ops,** should be identical, or very similar, centered on **version control repositories**. This approach upholds the principle that **the repository is the only source of truth**. In the operations world, this is known as **GitOps** (a concept created by Weaveworks), due to the popularity of GitHub, and it requires the desired state or operation on the system to be stored such that authorized readers can view the entire audit trail of changes. All changes to the desired state are fully traceable commits associated with committer information, commit IDs, and time stamps. This means that **the infrastructure is now a set of versioned artifacts** and can be audited using the standards of software development and delivery. Find in the table below some examples of why choose this operating model.

#	GITOPS VALUE	DESCRIPTION
1	**Visibility | Self-documenting Operations (Audit & History)**	Every change to any environment must happen through the repository. It's always possible check out the master branch and get a complete description of what is deployed where plus the complete history of every change ever made to the system. With this action an audit trail is retrieved with all changes in the system.
2	**Collaboration | Shared Knowledge in Teams**	Using a version control system to store complete descriptions of the deployed infrastructure allows everybody in the team to check out its evolution over time. With good commit messages, everybody can reproduce the thought process of changing infrastructure and easily find examples of how to set up new systems.
3	**Efficiency | Simpler Error Recovery**	With a complete history of how the environment changed over time, error recovery can be simpler - as issuing a git revert and watching the environment restored. The Git record is then not just an audit log but also a transaction log. It's possible can roll back & forth to any state and optimize the mean time to recovery (MTTR).
4	**Security | Safer Operations Management**	GitOps allows to operate completely inside a system (e.g., Git), with Operations teams/admins without direct access to the environments/resources. The pipelines can have all the security controls required before applying changes.
5	**Security | Operational Security**	GitOps improves security in the code & configurations (access, versioning, change, automated scans, and audit) and how changes are made to production.

The GitOps model can be used with two operational patterns:

- **Declarative Operations** instead of focusing on the steps to achieve the end goal focuses on specifying the end goal, the desired final state (i.e., target infrastructure). Through a purpose-built program, it is interpreted to realize it. With Terraform this goal is defined in the **tfvars** files and will be used to create and update infrastructure in OCI according

to the target state. **The OCI Open LZ focuses on this model**, using terraform code repositories and operations repositories for versioning the **tfvars.**

- **Procedural Operations** focus on operating **playbooks** that are used to interact with specific resources and their lifecycle or configurations. It's a classic process, with a set of activities required to achieve the goal. These playbooks and related configurations (one playbook with the generic code to N configurations) should be versioned on repositories. There are several technologies for this, being Ansible a popular one. This type of operation is compatible with the target operating model present in the Operational Model, using generic ansible code repositories and operations configuration repositories for versioning the specific configuration**.** For extending the OCI Open LZ with this type of operation refer to Design Considerations decision D14.

The diagram below presents the elements that normally participate in a GitOps operating model, using declarative operations with Terraform or procedural operations with Ansible.

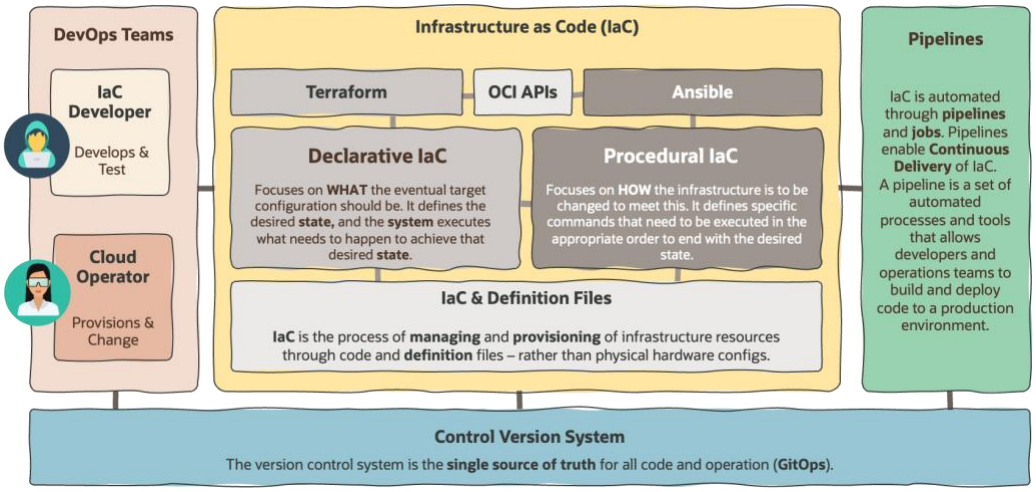


Figure 20 Operations View – Elements of a GitOps operating model.

The diagram below presents a runtime view of these elements, and the following table describes those elements and presents the OCI Open LZ runtime technology decision - in the last column. These decisions are made to deliver a tangible OCI Open LZ runnable solution in the next chapter (refer to the Runtime View), and do not represent a prescription or a recommendation, they are just a reference that can be reviewed and used and adjusted toward different technologies. The OCI Open LZ pattern can be replicated or adjusted to use other versioning control or CI/CD systems.

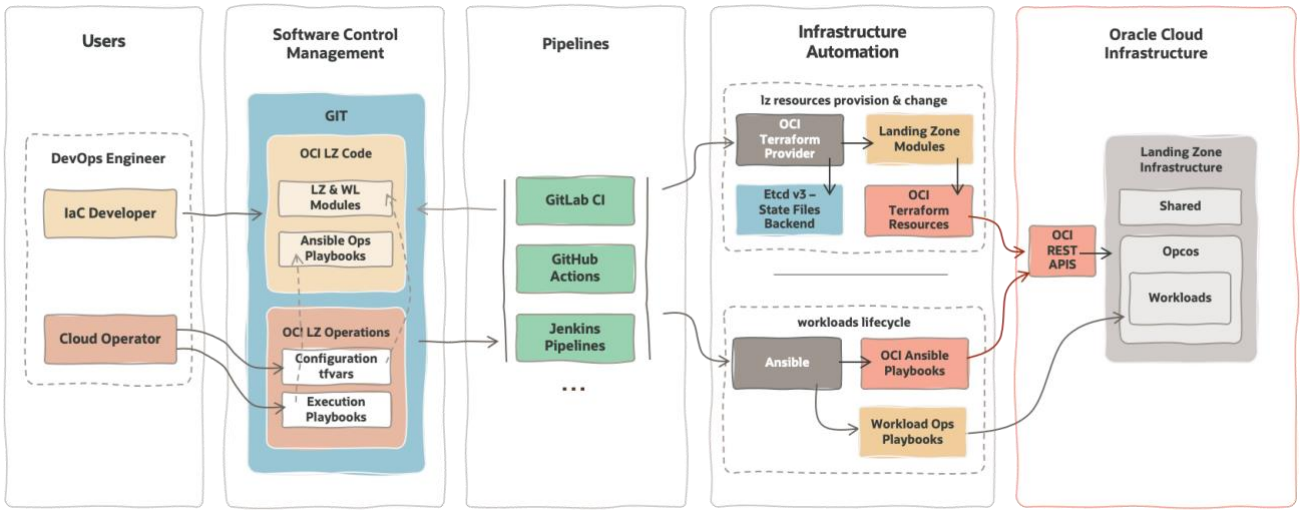


Figure 21 Operations View – GitOps runtime elements.

ELEMENTS	SUB-ELEMENT	DESCRIPTION	OPEN LZ RUNTIME
Personas	**IaC Developers**	The developers of the IaC that is responsible for the provision of the Landing Zone resources.	--
	Cloud Operators	A shared services team will operate the shared elements of the landing zone. Each OE will have its cloud operators.	--
Software Control Management	**Landing Zone Code Repository**	Source repositories with the Landing Zone Infrastructure as Code (IaC).	GitHub
	Landing Zone Operations Repository	Source repositories with the Landing Zone Operations configurations.	GitHub
Pipelines	**Pipelines and Jobs**	Technology that leverages pipelines to automate operations scenarios, triggered by commits.	GitHub Actions
Infrastructure Automation - Terraform	**OCI Terraform Provider**	Technology that allows the use of Terraform to interact with OCI resources.	--
	Landing Zone Configuration	The configurations that are being deployed for each scenario, as input to the landing zone code.	Declarative Ops
	Landing Zone Code	The Landing Zone IaC runs operations configurations.	GitHub
	OCI Terraform Resources	OCI Terraform modules to manage OCI resources.	--
	State Files Backend	Technology to manage the Terraform state file backend.	OCI Object Storage
	Ephemeral Virtual Machine	The VMs where the pipeline jobs will run. Depicted with the dotted squares.	GitHub Actions
Infrastructure Automation - Ansible	**Ansible**	Ansible enables IaC and it is used for provisioning, configuration management, and application lifecycle operations.	--
	OCI Ansible Playbooks	Oracle provides several Ansible modules to interact with OCI. More details are here and here.	--
	Workload Operations Playbooks	Custom Ansible modules to operate the target workloads.	--
OCI	**OCI REST API**	OCI REST API will be used by OCI Terraform Resources to create OCI resources.	--

	OCI Tenancy	OCI Target Tenancy.	--

Provisioning and Change

The objective of this section is to present the most significant elements that are part of the provisioning and change of cloud resources. These are divided into three areas:

- **The What:** The operation **scenarios,** or the most significant operation use cases associated with the creation or update of a set of resources.
- **The Who:** The **actors** that are responsible for the execution of those operations scenarios.
- **The How**: How the operation scenario will be **automated.**

Cloud Operations Teams (Who)

It's important to remember that the OCI Open LZ pattern has a **distributed operating model** with two types of teams:

- A **central team** (CT_OPS) is responsible to create and operate the core landing zone resources, known as shared services, and onboarding Operating Entities (OE).
- The **OEs operations teams** (OE_OPS) , there will be one per OE, will have the autonomy to operate (create and change) their resources on top of standard pre-defined structures.

These teams will operate the provisioning of resources for landing zone and OE-related workload tasks, and they will be members of multiple IAM groups to fulfill their tasks.

#	ACTORS	DESCRIPTION	OCI GROUPS
CT_DEV	**Central IaC Development Team**	• Responsible to create and maintain IaC and code repository for operational use.	Out of scope.
CT_OPS	**LZ Central Operations**	• Responsible to create and operate the core landing zone resources in the OCI Tenancy, known as shared services, and onboarding Operating Entities (OE) • This team will have tenancy admin roles. • Responsible for the central operations repository. • Responsible for updating Git repositories on core landing zone resources (network and security).	**GRP.01 to GRP.07** (View Groups)
OE_OPS	**Operating Entity Operations**	• There will be one OE_Ops per OE onboarded. • Each OE_Ops will have the autonomy to operate (create and change) their resources on top of standard pre-defined structures. • If applicable, these groups can create sub-groups to manage production and non-production environments, departments, DevOps, etc. For the sake of simplicity, these elements are not considered in this design. • Responsible for the OE operations repository and updating OE dedicated resource configuration on it. • Responsible for operating the future workloads on their landing zone area.	**GRP.OE.01** **GRP.OE.02** **GRP.OE.03** **GRP.OE.04** (View Groups)

The **LZ Central Operations team** provisions all related resources for the tenancy. This includes initial tenancy configuration (one-off operation) and shared services and OE initial structure.

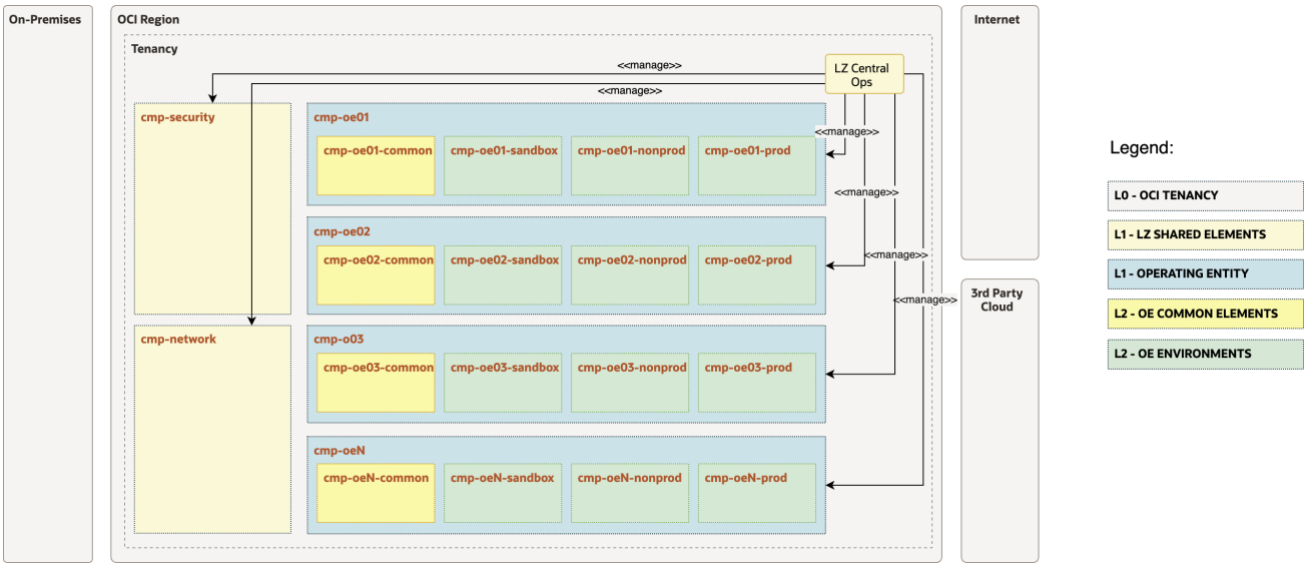


Figure 22 Operations View – Central Operations resource responsibilities.

The **OE Operations Team** provisions and modifies all the required resources within the OE compartment structure. It is a member of all related IAM groups. See the example below for **OE01.**

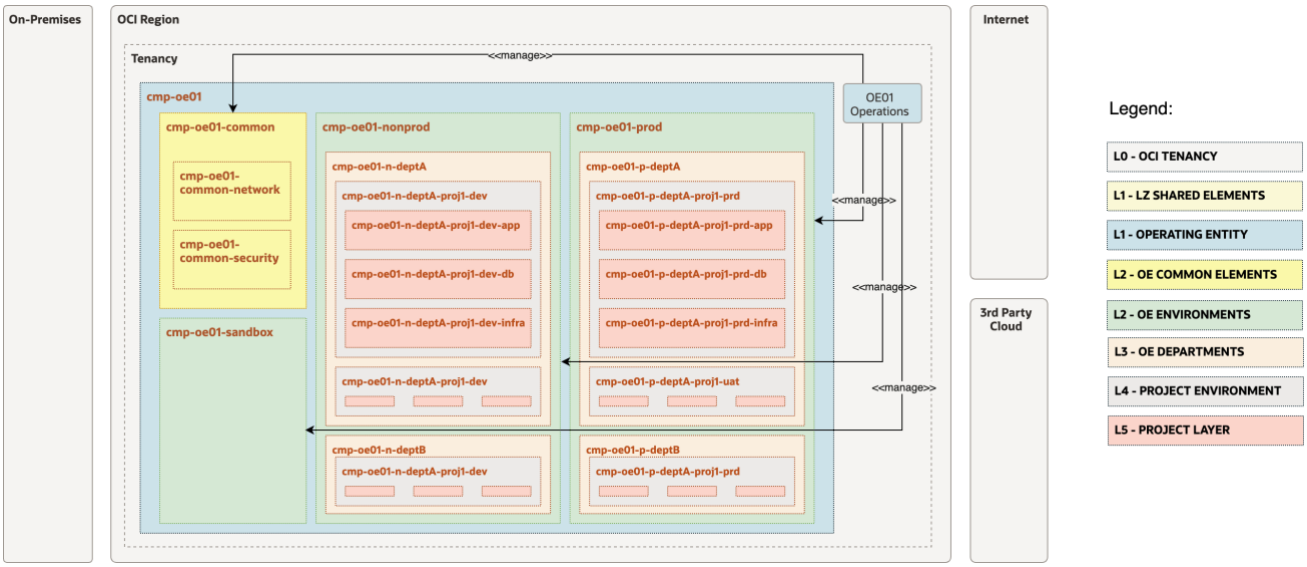


Figure 23 Operations View – OE Operations resource responsibilities.

Note that the decision for splitting the CT_OPS and OE_OPS operational scope is to demonstrate an example of separation of duties. In other possible scenarios, the OCI Open LZ can be managed by only CT_OPS, or even by more teams than the current two types proposed. With this example, it's possible to evolve or tailor the operating model toward the organizational needs.

Operations Scenarios (What)

The **operations scenarios are one of the most important elements of this design**, as they represent the **use cases** and its key operations activities on the OCI Open LZ that create or update resources. An operation scenario is normally triggered by a service request, on a ticketing system.

An operations scenario in a more formal definition should be seen as an **operational process**, which is a set of correlated activities executed as one **unit of work,** with its own **frequency.** A scenario can contain one to several **activities**, and the level of automation may vary depending on each target system. Each operation scenario has also an owner, which will be responsible for its execution. The owner will be the operations team which has associated OCI Groups and policies that allow the management of those resources.

Find below the operations scenarios identified for the OCI Open LZ. Note that some of these due to the separation of duties between the CT_OPS and OE_OPS will interlock, for example, after setting up a new OE project (OP.04), some network updates will be required for the project to be connected, such as OE VCN/Subnets Route Tables (OP.02), DRG rules (OP.01), among other resources. The interlock column captures this type of dependency.

OP.01 – Manage Shared Services

DESCRIPTION	Creates or changes the shared elements of the landing zone and applies posture management.
OWNER	CT_OPS
PRE-REQUIREMENTS	Tenancy onboarded and one-off configurations executed.
SECURITY RESOURCES	Compartments, Groups, Policies
NETWORK RESOURCES	DRG, VCN, Subnets, SL, RT, DRG Attach., Network Firewall, DNS
TEMPLATE	View Runtime Example
INTERLOCK	N/A
# EXECUTIONS	N per Tenancy
FREQUENCY OF USE	High

OP.02 – Manage OE

DESCRIPTION	Onboards or changes an OE, creating the OE structures that will be used by the OE to create resources. This operation when executed the first time, onboards an OE. This first execution requires a set of activities to prepare an OE Operations team for running their operation scenarios (e.g., handover the OCID for OE core resources). The pipelines section and design consideration "D11.5 Simplify OE Onboarding" present how this type of process can be further simplified or automated.
OWNER	CT_OPS
PRE-REQUIREMENTS	OP.01 Manage Shared Services executed.
SECURITY RESOURCES	Compartments, Groups, Policies
NETWORK RESOURCES	VCN, Subnets, SL, RT, DRG Attachments, Service/Internet Gateways
TEMPLATE	View Runtime Example
INTERLOCK	N/A

# EXECUTIONS	N per Tenancy
FREQUENCY OF USE	High

OP.03 – Manage Department

DESCRIPTION	Creates and changes a new department structure to receive department projects.
OWNER	OE_OPS
PRE-REQUIREMENTS	OP.02 CT_OPS hand-over the OCIDs for the project parent compartments and project VCNs.
SECURITY RESOURCES	Compartments, Groups, Policies
NETWORK RESOURCES	--
TEMPLATE	--
INTERLOCK	OP.02 (Security Updates)
# EXECUTIONS	N per OE
FREQUENCY OF USE	Low

OP.04 – Manage Project Environment

DESCRIPTION	Creates or changes a project with the related environments and application layers.
OWNER	OE_OPS
PRE-REQUIREMENTS	OP.03 CT_OPS shares OCIDs for the project parent compartments and project VCNs.
SECURITY RESOURCES	Compartments, Groups, Policies
NETWORK RESOURCES	NSG
TEMPLATE	--
INTERLOCK	OP.02 (Network Updates)
# EXECUTIONS	N per Department
FREQUENCY OF USE	High

OP.05 – Manage PoC Project

DESCRIPTION	Creates or changes a PoC project in the OE Sandbox environment.
OWNER	OE_OPS
SECURITY RESOURCES	Compartments, Groups, Policies

NETWORK RESOURCES	Subnet
TEMPLATE	--
INTERLOCK	OP.02 (Security & Network Updates)
# EXECUTIONS	N per OE
FREQUENCY OF USE	MEDIUM

Automation (How)

The key responsibility of the cloud operator (OE_OPS and CT_OPS) is creating two configuration files - with **tfvars/json** format - for each operation scenario, one security configuration, and one network configuration. The component model below presents the operator's direct responsibilities in the automation of the operation scenarios, and the Terraform modules that will, directly and indirectly, provision and change resources. This solution simplifies the OCI Open LZ operations automation in the following way:

- The cloud operator focuses only on **human-readable JSON** format **configuration.**
- There will be **templates and examples** to speed up any configuration, in the **terraform-oci-open-lz-templates** repository. This repository is owned by Operations teams and contains all key operations scenarios as configurations.
- There is only **one façade module** as a direct interface for operations, the **Orchestrator** module in the **terraform-oci-open-lz** repository. The façade module will hide the technical orchestrator of resources using four modules in dedicated repositories, to create compartments, groups, policies, and networking, respectively. These repositories are owned by Development teams.
- To automate an operation procedure, upon a **service request (0),** there are two options:
    - If it's the **creation of new OCI resources,** the Cloud Operator will **retrieve an existing template for the specific operation (1), update it with the specific values (2),** and finally **execute terraform plan/apply** on the **Orchestrator,** with the **configurations (3).** Refer to the following diagram for the visual flow.
    - If it's the **change of existing resources**, the process will be like the step above, by directly changing the existing configuration file, without use of templates as the resources were already configured and instantiated.
- Note this is a simple perspective of how to automate operations. In the next section a target operating model is proposed to govern the lifecycle of all operations.

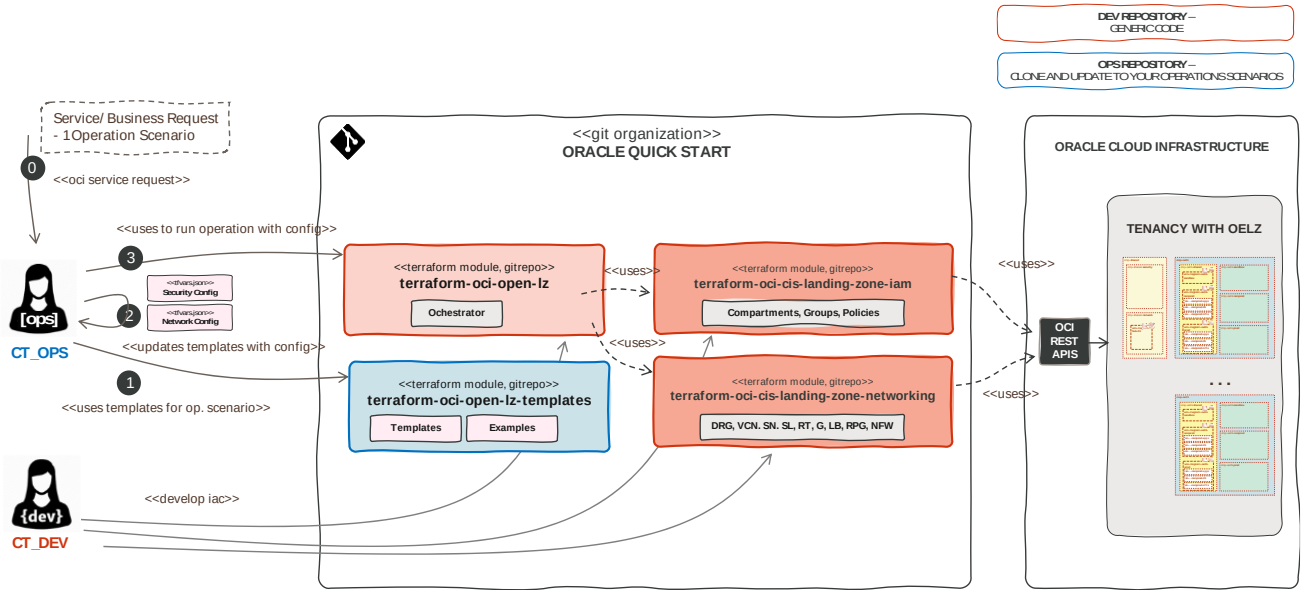


Figure 24 Operations View – Component model for a simple automation scenario.

The table below maps the operations scenarios to Terraform modules, with sample configurations that implement the provisioning and change of resources - enabling the declarative operating model. In the diagram above the CIS Landing Zone Enhanced Modules are indirectly used to solve the **Identity and Access Management** (IAM) [REF.09] and **Network** resources [REF.10], nevertheless, this is merely illustrative, like all the OCI Open LZ, any other set of modules can be used to solve the automation. Note the façade modules are also optional, depending on your approach to orchestrating terraform and managing templates.

In the diagram above the CIS Landing Zone Enhanced modules [REF.12] – directly used by the OCI Open LZ Orchestrator [REF.00] – are used to solve the **Identity and Access Management** (IAM) [REF.09] and **Network** resources [REF.10]. Nevertheless, this is merely illustrative, like all the OCI Open LZ, any other set of modules can be used to solve the automation. Note the façade modules are also optional, depending on your approach to orchestrating terraform and managing templates.

For further automating and expanding the landing zone with more resources, please refer to the CIS Landing Zone Enhanced Modules [REF.12] for more capabilities on security, observability, and governance.

Operating Model

The previous section introduced the **actors** (who) that will execute each **operation scenario** (what) to operate the OCI Open LZ with a declarative approach using Terraform. This section introduces the **OCI Open LZ GitOps operating model,** a key element to enable continuous declarative operations using Git repositories as the **single source of truth** for OCI Open LZ operations. This operating model is ready for **day 2 operation** on the OCI Open LZ.

As stated in Cloud-Native Operations section, the OCI Open LZ follows a Git-centric solution to be tangible, nevertheless, **this pattern can be adjusted towards another versioning control system and orchestration technologies.**

The following diagram presents a set of artifacts involved in this operating model, the Git **repositories**, and the **pipelines** to automate each operations scenario.

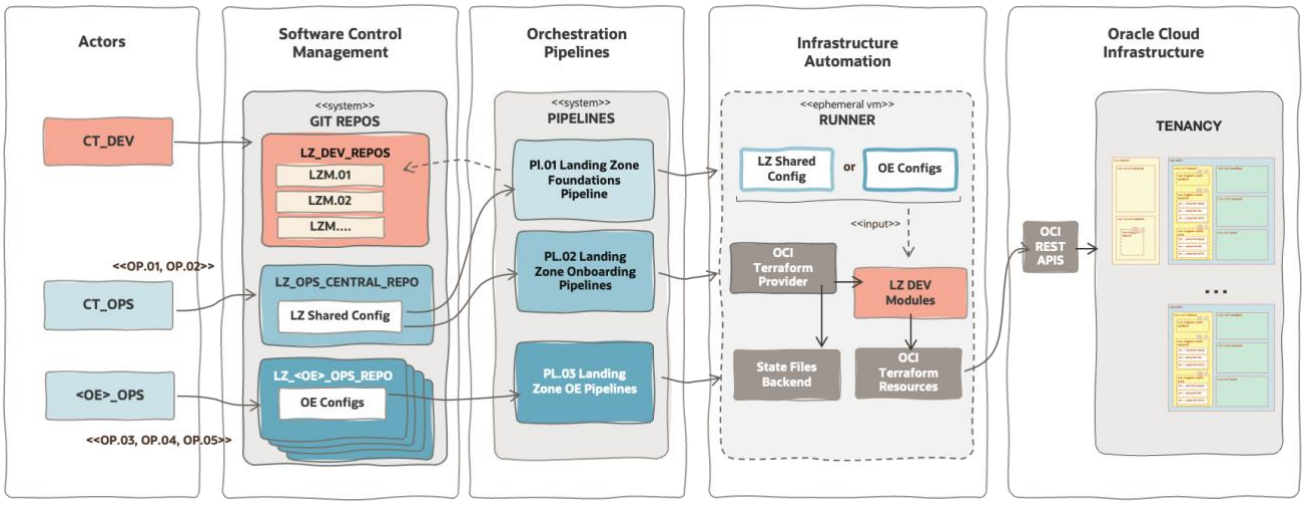


Figure 25 Operations View – Technical cloud operating model.

Note that there will be two types of repositories:

- The **operations repositories** will contain only IaC configurations.
- The **code repository** will be used to run these configurations repeatedly. The lifecycle of this repository is out of the scope of this document.

This separation is crucial and is depicted in the table below, presenting the several repositories that are part of the OCI Open LZ pattern.

ID	REPOSITORY	TYPE	OWNER	DESCRIPTION	OPS

REP.1	**DEVELOPMENT REPOSITORIES -** "LZ_DEV_REPOS"	DEV	CT_DEV	• These repositories contain all the IaC used to run any OCI Open LZ operations scenario.
				• The OCI Open LZ uses generic code that can be used for other types of Landing Zone solutions. It contains:
				  ○ Four domain-driven, self-contained modules for compartments, groups, policies, and networking.
				  ○ One orchestration module will act as a façade to the four domain modules, distributing the configurations toward them. Its main objective is to create a scope of resources that enables all four configurations to be related in the Terraform dependency graph.
				• Each operation scenario will have its specific configurations, that will be input to the core orchestration module.
				• It's recommended to split the code repositories into two organizations:
				  ○ Terraform Modules Organization will contain all four modules and future modules at the same functional level.
				  ○ QuickStart Organization will contain modules that will be used to simplify the use of the domain modules as the first interface.
				• The code repositories and git organizations are depicted in the automation section.
				• A git commit on this type of repository will launch the pipelines to manage the Landing Zone Software Development Life Cycle (SDLC).
					N/A

REP.2	**CENTRAL OPERATIONS REPOSITORY -**	OPS	CT_OPS	• There will be one LZ_OPS_CENTRAL_REPO. Note that depending on how distributed the operations of the Landing Zone are, there can be one single repository containing all regional configurations as	OP.01 OP.02

	"LZ_OPS_CENTRAL_REPO"			substructures or several repositories - one per region, operated by different sub-teams. • This repository contains all the tfvars/json configuration files for the core landing zone resources. • This repository will also contain the central repository workflows/pipelines to run the operations scenarios. • A git commit will launch the automation pipelines for operations.	
REP.3	**OE OPERATIONS REPOSITORIES -** "LZ_OPS_<OE>_REPO"	OPS	OE_OPS	• There will be several LZ_OPS_<OE>_REPO, one per OE. Note that depending on how distributed the operations of an OE is, an OE can have one single repository containing all regional configurations as substructures or several repositories - one per region, operated by different sub-teams. • This repository contains all the tfvars/json configuration files for the OE resources. • This repository will also contain the OE repository workflows/pipelines to run the operations scenarios. • A git commit will launch the automation pipelines for operations. • Note that if an OE_OPS team doesn't have Terraform configurations skills, they can delegate this configuration to the CT_OPS team. In this case, CT_OPS can centralize OE operations also, or just operate some OEs. For these cases, the LZ_OPS_CENTRAL_REPO can version all the OE-related configurations for all operations scenarios.	OP.03 OP.04 OP.05
REP.4	**TEMPLATES REPOSITORIES -** "LZ_OPS_TEMPLATES_REPO"	**OPS**	**CT_OPS**	• Templates are required to simplify the cloud operations activities, reducing efforts and possible errors in configurations. • Each operation scenario will have related templates in the form of tfvars/json files, with an existing JSON structure that will reflect the resources for each operation scenario. • The cloud operator uses these templates directly, replacing the necessary values., and versioning them on REP.2 and REP.3 repositories. • Templates can be stored on an independent repository, as they have their lifecycle, distributed by several repositories according to scope, or even be part of each specific operations repository (REP.2 and REP.3). This decision will depend on how centralized or distributed the template management is.	ALL

Operation Pipelines

The table below suggests possible automation pipelines to automate the operations scenarios using webhooks to Git operations repositories (REP.2 and REP.3). There are plenty of solutions to solve this operational area, which will differ from customer to customer. One important characteristic is where the pipeline's job will run. To keep the option simple and in line with the type of customer using an OCI Open LZ, we can consider two options on where to run the pipeline OCI jobs:

1. On the native platform targeting OCI remotely
2. Inside OCI, with an ephemeral compute instance. For example, GitLab allows configuring where the jobs will run whereas GitHub actions do not allow this.

This decision will require the configuration the pipeline/job service user in different ways as depicted in the table below.

PL.01 – Central Pipeline

DESCRIPTION	• Automates the provisioning and change of core landing zone resources. • Automates the onboarding process and change process of an OE. • In the OCI Open LZ runtime, this pipeline will run terraform plan and terraform apply. With the GitHub Actions implementation, this pipeline will have two workflows for this separation. • Refer to the Design Considerations section (D11.5 Simplify OE Onboarding) for further extending this pipeline for more automation.
OWNER	OE_OPS
OCI JOB OUTSIDE OCI	Service User | part of CT_OPS
OCI JOB INSIDE OCI	Job uses Instance Principal (Dynamic Group at Root Level)
OPERATIONS SCENARIOS	OP.01, OP.02
RELATED REPOSITORIES	Set on REP.2 Uses REP.1

PL.02 – OE Pipeline

DESCRIPTION	Automates the provisioning and change of OE dedicated resources. As this element stands at the OE level, an OE can have two options to implement this: • leverage potentially existing corporate technology to execute this pipeline. • use their technology to automate operations in their OE area. For example, in the case that an OE is an OpCo, this operating company will most probably leverage its existing technology to use OCI. In summary, this element will exist, and where it will be implemented might diverge from customer to customer. In the OCI Open LZ runtime, this pipeline will run terraform plan and terraform apply. With the OCI Open LZ GitHub Actions implementation, this pipeline will have two workflows for separating activities.
OWNER	OE_OPS
OCI JOB OUTSIDE OCI	Service User | part of OE_OPS
OCI JOB INSIDE OCI	Job uses Instance Principal (Dynamic Group at OE Level)
OPERATIONS SCENARIOS	OP.03, OP.04, OP.05
RELATED REPOSITORIES	Set on REP.2 Uses REP.1

Central Repository Structure (REP.2)

Find below an example of what could be a git repository structure for the core landing zone resources. In this example, there are two OEs (ZYZ and JKL). The first column's color code matches the ERD's color code in the Functional View and the architecture diagrams presented in the Security View and Network View.

REP.2 | LZ CENTRAL REPO STRUCTURE	CONFIGURATION SCOPE	OP SCENARIOS	PIPELINES
LZ_OPS_CENTRAL_REPO		--	--
LZ_OPS_CENTRAL_REPO/WORKFLOWS	**Pipelines**		
LZ_OPS_CENTRAL_REPO/**SHARED**	**tfvars/json** files to create the security and networking elements on the shared structure.	OP.01 Manage SS	--
LZ_OPS_CENTRAL_REPO/OE/**OE_XYZ**	**tfvars/json** files to create the OE_XZY 3 environments and common elements	OP.02 Manage OE	--
LZ_OPS_CENTRAL_REPO/OE/...	...	...	...
LZ_OPS_CENTRAL_REPO/OE/**OE_ABC**	**tfvars/json** files to create the OE_ABC 3 environments and common elements	OP.02 Manage OE	--

Figure 26 Operations View – Central Repository Structure.

OE Repository Structure (REP.3)

Find below an **example** of what could be a Git repository structure for an OE. In this example:

- The OE is called **OE_XYZ.**
- **OE_XYZ** has two **departments: DEP_A** and **DEP_B.**
- **DEP_A** has two projects: **PROJ1** and **PROJ2.**
- **DEP_A PROJ1** has 3 environments: development **(DEV),** production **(PRD)**, and disaster recovery **(DR).** The production environment is disclosed to exemplify the substructure of each layer.
- **DEP_A PROJ2** has 2 environments: development **(DEV),** and production **(PRD)**.
- **DEP_B** has only one project: **PROJ3.**
- **DEP_B PROJ3** has only one environment **(DEV).** The development environment is disclosed to exemplify the substructure of each layer.

The first column's color code matches the ERD's color code in the Functional View and the architecture diagrams presented in the Security View and Network View.

REP.3 | OE REPO STRUCTURE	CONFIGURATION SCOPE	OP. SCENARIOS	PIPELINE
LZ_OPS_OE_XYZ-REPO	**OE_XZY** Repository	--	--
LZ_OPS_OE_XYZ-REPO/**WORKFLOWS**	OE_XZY **Pipelines**	--	PL.02
LZ_OPS_OE_XYZ-REPO/**PROD**	**OE_XZY** Production configurations	--	--
LZ_OPS_OE_XYZ-REPO/PROD/**DEP_A**	**DEP_A tfvars/json** files for production resources	OP.03 Department	--
LZ_OPS_OE_XYZ-REPO/PROD/DEP_A/**PROJ1**	**DEP_A** Resources for **PROJ1** production environments	--	--
LZ_OPS_OE_XYZ-REPO/PROD/DEP_A/PROJ1/**PROJ1-PRD**	**tfvars/json** files to configure the **PROJ1** production resources.	OP.04 Project Env.	--
LZ_OPS_OE_XYZ-REPO/PROD/DEP_A/PROJ1/PROJ1-PRD/**APP**	**tfvars/json** files to configure the **PROJ1** production application resources.	*Out-of-scope*	--
LZ_OPS_OE_XYZ-REPO/PROD/DEP_A/PROJ1/PROJ1-PRD/**DB**	**tfvars/json** files to configure the **PROJ1** production database resources.	*Out-of-scope*	--
LZ_OPS_OE_XYZ-REPO/PROD/DEP_A/PROJ1/PROJ1-PRD/**INFRA**	**tfvars/json** files to configure the **PROJ1** production infra resources.	*Out-of-scope*	--
LZ_OPS_OE_XYZ-REPO/PROD/DEP_A/PROJ1/**PROJ1-DR**	**tfvars/json** files to configure the **PROJ1** disaster recovery resources.	OP.04 Project Env.	--
LZ_OPS_OE_XYZ-REPO/**NON-PROD**	**OE_XZY** Non-production configurations	--	--
LZ_OPS_OE_XYZ-REPO/NON-PROD/**DEP_A**	**DEP_A tfvars/json** files for non-production resources	OP.03 Department	--
LZ_OPS_OE_XYZ-REPO/NON-PROD/DEP_A/**PROJ1**	**DEP_A** Resources for **PROJ1** non-production environments	--	--
LZ_OPS_OE_XYZ-REPO/NON-PROD/DEP_A/PROJ1/**PROJ1-TST**	**tfvars/json** files to configure the **PROJ1 test** resources.	OP.04 Project Env.	--
LZ_OPS_OE_XYZ-REPO/NON-PROD/DEP_A/PROJ1/**PROJ1-STG**	**tfvars/json** files to configure the **PROJ1 staging** resources.	OP.04 Project Env.	--
LZ_OPS_OE_XYZ-REPO/NON-PROD/DEP_A/**PROJ2**	**DEP_A PROJ2** Resources for non-production environments	--	--
LZ_OPS_OE_XYZ-REPO/NON-PROD/DEP_A/PROJ2/**PROJ2-TST**	**tfvars/json** files to configure the **PROJ2** test resources.	OP.04 Project Env.	--
LZ_OPS_OE_XYZ-REPO/NON-PROD/DEP_A/PROJ2/**PROJ2-STG**	**tfvars/json** files to configure the **PROJ2** staging resources.	OP.04 Project Env.	--
LZ_OPS_OE_XYZ-REPO/NON-PROD/**DEP_B**	**DEP_B tfvars/json** files for non-production resources	OP.03 Department	--
LZ_OPS_OE_XYZ-REPO/NON-PROD/DEP_A/**PROJ3**	**DEP_B** Resources for **PROJ3** non-production environments	--	--
LZ_OPS_OE_XYZ-REPO/NON-PROD/DEP_A/PROJ1/**PROJ3-DEV**	**tfvars/json** files to configure the **PROJ3** development resources.	OP.04 Project Env.	--
LZ_OPS_OE_XYZ-REPO/**DEVELOPMENT**	**OE_XZY** development configurations	--	--
LZ_OPS_OE_XYZ-REPO/DEVELOPMENT/**DEP_A**	**DEP_A tfvars/json** files for development resources	OP.03 Department	--
LZ_OPS_OE_XYZ-REPO/DEVELOPMENT/DEP_A/**PROJ1**	**DEP_A** Resources for **PROJ1** development environments	--	--
LZ_OPS_OE_XYZ-REPO/DEVELOPMENT/DEP_A/PROJ1/**PROJ1-DEV1**	**tfvars/json** files to configure the **PROJ1 dev1** resources.	OP.04 Project Env.	--
LZ_OPS_OE_XYZ-REPO/**SANDBOX**	**OE_XZY** sandbox configurations	--	--
LZ_OPS_OE_XYZ-REPO/SANDBOX/**PROJ5**	**tfvars/json** files to configure the **PROJ5 sandbox** resources.	OP.05 PoC Project	--

Figure 27 Operations View – OE Repository Structure.

Templates Repository Structure (REP.4)

The table below presents an example of how to organize **configuration templates** in a Git repository, following a similar structure used on the operations repository. The first column's color code matches the ERD's color code in the Functional View and the architecture diagrams presented in the Security View and Network View.

REP.4 | TEMPLATES ORGANIZATION	TEMPLATE SCOPE	OPERATION SCENARIOS
LZ_OPS_TEMPLATES/**WORKFLOWS**	OCI Open LZ **Pipelines**	
LZ_OPS_TEMPLATES//**SHARED**	**tfvars/json** template to configure the **shared** structures	OP.01 Shared Services
LZ_OPS_TEMPLATES/**OE**	**tfvars/json** template to configure the **OE** structures (Environments, Common, etc.)	OP.02 Manage OE
LZ_OPS_TEMPLATES/OE/**DEP**	**tfvars/json** template to configure the **department** structures	OP.03 Manage Department
LZ_OPS_TEMPLATES/OE/DEP/**PROJ1**	**tfvars/json** template to configure the **project** structures	OP.04. Manage Projects

Figure 28 Operations View – Templates Repository Structure.

Modus Operandi

For operating OCI Open LZ with **Git Repositories** the **Operations team** (CT_OPS or OE_OPS) will use a repeatable and versioned *modus operandi*, and for every **create request** (normally arriving through a ticket system or a service catalog) the Operator executes the following steps:

STEP	DESCRIPTION
0	A **service request** arrives at the Cloud Operator. All requests will match an operation scenario. Every operations scenario has a template.
1	Create a **new branch** with associated new folders in the target operations repository (REP.2 or REP.3)
2	**Copy the resource template files** from a secured and controlled git repository location (REP.4) to the target Operations repository structure (REP.2 or REP.3).
3	**Rename the templates and update the configuration variables** (e.g., project name) on the **tfvars** files (network and security) leveraging existing pre-determined variables and updating the required ones. **Perform a commit**, which will download from LZ_DEV_REPO (REP.1) the necessary Terraform modules to execute the automation process. In this OCI Open LZ pattern, LZ_DEV_REPO (REP.1) contains generic Terraform code to run any configuration, therefore one plus four modules will implement all operations scenarios. Each operation scenario will have its specific configurations. The continuous delivery controls (module integration, syntax check, policy convention testing, etc.) in the pipeline can run, finishing with a terraform plan.
4	**Create a Pull request.** Optionally a member of CT_OPS or OE_OPS can review the previous commit, review the results of the controls, and terraform plan, and approve the change to be merged into the main branch. The **merge** into the main branch fires a **commit.**
5	The **Git commit will launch the associated pipeline**, performing the terraform **apply** and doing the needed deployment/modifications in OCI.

For an **update request** step 2 is not necessary as the configurations already exist in the target operations repository and updating the configuration variables in step 3 will be the main step. Steps 1, 4, and 5 will remain equal.

The diagram below presents a high-level example component model for the CT_OPS operations, with the five steps above and key artifacts used: configurations, git repositories, and Terraform modules. The five steps flow is an example of the OP.02 Managing an OE. Note that the configurations on LZ_OPS_CENTRAL_REPO will match the resources on the OCI Open LZ tenancy.

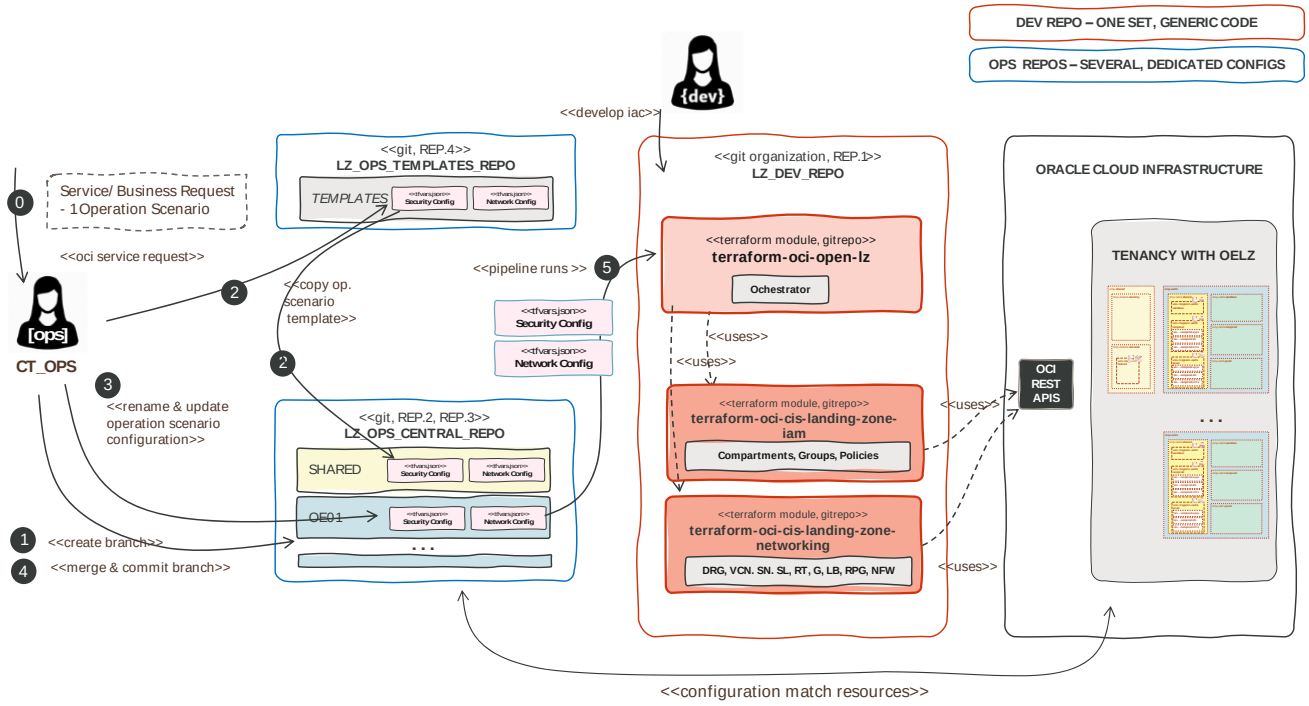


Figure 29 Operations View – Modus operandi.

As stated at the beginning of the chapter, this approach upholds the principle that **Git is the only source of truth**. The operation scenarios executed on the system are stored in Git such that authorized readers can view the entire audit trail of changes. All changes to the desired state will be fully traceable with commits associated with committer information, commit IDs, and time stamps. This means that **OCI Open LZ infrastructure is a set of versioned artifacts** and can be audited using the standards of software development and delivery.

Design Considerations

The previous chapters presented a proposal to automate and operate the OCI Open LZ using a GitOps approach. There are several alternatives to this model, more granular and less granular. Consider the following topics when adjusting or extending the OCI Open LZ.

TOPIC	ID	DECISION	GUIDELINES
Granularity of Configuration Files	**D1**	**Use Macro-segmentation**	• There are several possibilities for this option, such as one configuration file per OE, or even one configuration file for the OCI Open LZ, among others. • Aggregating several configurations into a few files will most probably drive the OCI Open LZ operations to an unmanageable state in the near term. There are plenty of drawbacks to this solution. Since all resources are in a few configuration files, consider the level of risk associated with any update, by changing unwanted resources, the time it can take to process these changes, the locking mechanism to manage state files that will block parallel changes on different type resource structures, and even the segregation of duties of the several possible operations teams. • This is not recommended.
	D2	**By Operations Scenarios**	• **This is the recommended approach** since it focuses on the user experience of the main actor, the cloud operator. The configurations also directly match an operation scenario that is scoped by design to a specific set of resources. • This enables the use of templates for each operation scenario, reducing effort and possible human errors. • One operation will never block or interfere with another operation running in parallel. • Since one operation scenario can target production or non-production, or other environments, this should be solved at the repository structure level.
	D3	**By Resource Domain**	• This approach focused on the typical segregation of duties, such as those associated with security, network, database resources, etc. • This approach should not be used as the single option to run operations, as it will mix resources from different types of areas, aggregating unrelated resources and therefore complicating the operations team activities as they will have service requests based on usage. • **This approach can be used to complement D2. In this OCI Open LZ pattern, we use D2 with D3 with security and network segmentation.**
	D4	**Micro-Segmentation**	• There are several possibilities for this option, such as one configuration file per resource type or even one file per resource. • Segmenting too much of a configuration file can lead to too many activities by the operations teams for one specific operation scenario. This ultimately increases efforts and the probability of operational errors. • This is not recommended.
Number of Repositories	**D5**	**One Operations Repository**	• Consider this option if there is **only one single operations team** that centralizes all the management of the OCI Open LZ.
	D6	**Several Repositories - By Operation Teams**	• In the OCI Open LZ, we use this solution because there is a clear separation of duties among the central operations team and the several OE operations teams.

	D7	**Several Repositories - Multi-cloud**	In case of facing a multi-cloud OCI Open LZ, targeting OCI and other CSPs, consider the following options in the case of D5 or D6: • **Separate the target CSP at the repository root level** (i.e., with folders OCI, GCP, AWS, etc.), and add the configurations associated with the functional organization of resources under these folders. This is the normal scenario where one team oversees all multi-cloud operations. • **One repository per each CSP** and try to have a common or similar repository structure for all, to have homogeneous GitOps operations among CSPs. In this scenario, the repositories' structure can be similar but there is a hard operational separation between cloud providers. This is normally the case, but not limited to when there are dedicated teams to operate each cloud provider and they're not supposed to see/use/change each other resources.
	D8	**Several Repositories - Regions and DR**	• Use the same criteria as D7, and instead of CSPs there will be OCI Regions.
Granularity of the State File	**D09**	**By Operations Scenarios**	• **This is the recommended approach.** In the OCI Open LZ each Terraform command (plan and apply) will be executed with the two **tfvars.json** files (security and network) associated with a specific scope (in the Git repository), which maps to a specific resources/operations scenario (OP.X). Therefore, the OCI Open LZ state files will contain the complete set of security and network resources associated with an operation scenario. • Consider using a backend that supports the locking of state files. This prevents any automation from acquiring the lock and potentially corrupting the state while in the middle of an operation. The OCI Open LZ design already isolates OEs, Departments, and Project Environments, nevertheless locking the state on any operations scenario is considered a best practice.
	D10	**Use Micro-Segmentation**	• A possible lower level of isolating state files would be by resource domain, i.e., security or network resources. This would require the execution of two Terraform commands (plan and apply) per resource domain, instead of one. Segmenting further inside the domain of the resources will increase drastically the number of Terraform commands per operations scenario. • Keeping micro-segmenting the state files will increase the operational complexity and will break the Terraform dependency graph, making the operations team responsible for manually associating resources between tfvar.json files. • This is not recommended unless there are hard operational requirements for different operations teams to be responsible for their resources, splitting an operation scenario into several sub-activities executed by several operations teams - at Terraform level.
Operational Automation	**D11**	**Reducing Human Effort**	The modus operandi section presents a set of activities executed by the cloud operations teams. These are tasks that a small and skilled operations team can master with low effort. Nevertheless, parts of these activities are repetitive and can be further automated with the support of other systems. While taking this decision, consider the following considerations. 1. **Data Input Validation:** automating tasks requires optimal input for all required parameters to avoid exceptions or blocked processes - as these will consume more time than a manual task itself. This required in-place validation logic on the target system. 2. **Business Self-Service:** providing business users the capability of requesting resources and having them provisioned or changed automatically is a powerful mechanism. Nevertheless, this mechanism it's hardly a plug & play as it needs to be aware of when it can run without human intervention, or with human intervention, and needs the operations scenario to be coded end-to-end to maintain the version of the operation configurations in the Git Repositories. 3. **Reduce Manual Tasks:** See D12 and D13 as examples of approaches to reduce human efforts and mitigate the risks of operational errors. 4. **Workloads:** See D14 as an example to expand this operating model to workload configurations and lifecycle. 5. **Simplify OE Onboarding:** To automate further the OE onboarding process consider complementing the PL.01 pipeline with another dedicated pipeline to onboard an OE. This new pipeline will run only once per OE, the first time OP.02 is executed. This pipeline can contain several activities related to the OE team and OE resources, such as the creation of the OE dedicated repository (REP.3), its Git users, adding OE templates prepared with the OCIDs from the OE resources they can use (not manage) so that an OE can start creating projects on top of their dedicated resources in no time, among any required activities that will reduce human error and OE onboarding time.

	D12	**Automating the "Modus Operandi"**	• A procedural approach can be used to execute partially or completely the human activities described on the modus operandi. For example, there can exist an Ansible configuration playbook for each operations scenario. The configuration and execution playbooks would also be versioned in the Git repositories, following the model proposed in this chapter. This approach would provide further levels of automation following the same - GitOps - operating model. • In this scenario, the operations teams would be focusing on executing versioned Ansible playbooks setting the required input, and monitoring operations, avoiding mainly the updates on several **tfvars/json** files.
	D13	**Using a Service Catalogue**	• Service Catalogs are often seen in enterprises that promote business self-service. Having some operations scenarios on a service catalog and requesting and validating the required input is a great pattern for successful automation. This service catalog can afterward execute the modus operandi logic internally if the system has capabilities for this or use a procedural automation mechanism such as the one described in D12.

	D14	**Expand the Operating Model to Include Workload Configurations and Lifecycle**	• The proposed operating model can be extended with procedural operations to cover the lifecycle of workloads - where a declarative model doesn't fit (e.g., patching operations).
			• To make a procedural model fit into the existing operating model it's recommended to use the same strategy for the declarative model, i.e., split the code repository and operations configurations repositories. The former will contain **Common Playbooks (CP)** and the latter **Execution Playbooks (EP).**
			  ○ **Common Playbooks (CP)**: These playbooks are the reusable common code for all the operations executed. An operational scenario can use one or more CP playbooks, and a playbook will orchestrate a set of activities to be performed at the workload level.
			  ○ The **CP** playbooks will be versioned in the code repository (REP.1) and will be used by the pipelines.
			  ○ **Execution Playbooks (EP):** These playbooks are dedicated to **operations scenarios** and will contain variables definitions/values and the main tasks, that will be used and implemented by **CP** playbooks, respectively.
			  ○ EP playbooks should be stored as templates in the REP.4.
			  ○ When an operation scenario is executed the **EP** playbook template will be used, updated, and versioned on REP.2 or REP.3 with the configuration for that scenario. The pipeline will use this EP and all the common CPs to run the operations.

Runtime View

This chapter is presented on the OCI Open LZ git repository [REF.11].