

Oracle® Database

Administering Oracle Blockchain Platform Enterprise Edition for Besu



G62105-01
September 2026



Oracle Database Administering Oracle Blockchain Platform Enterprise Edition for Besu,

G62105-01

Copyright © 2026, 2026, Oracle and/or its affiliates.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle®, Java, MySQL, and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>.

Oracle customer access to and use of Oracle support services will be pursuant to the terms and conditions specified in their Oracle order for the applicable services.

Contents

1	Oracle Blockchain Platform Enterprise Edition Overview	
	Deployment Model	4
	Required Infrastructure and Resources	5
	Getting Started with Oracle Blockchain Platform Enterprise Edition for Besu	7
2	Deploy Oracle Blockchain Platform Enterprise Edition for Besu	
	Deployment Prerequisites	2
	Deploy from Oracle Cloud Marketplace	3
3	Blockchain Platform Manager	
4	User Management and Identity Providers	
	User Management with OpenLDAP	1
	User Management with an External Identity Provider	2
	User Groups and Roles	5
5	Create a Rich History Database Configuration	
6	Create an Instance	
	Verify Your Instance	3
	Service Endpoints	4
7	Create an Organization and Wallet	
	Universal Wallet as a Service (UWaaS)	2
8	Manage the Organization and Network	

9	RPC Proxy	
10	Develop and Use Smart Contracts	
	One-Click Application Deployment	1
	Sample Smart Contracts	3
A	Application Information	
B	Connect to Oracle Kubernetes Engine	
	Index	

1

Oracle Blockchain Platform Enterprise Edition Overview

Oracle Blockchain Platform Enterprise Edition 26.3 for Besu provides a Kubernetes-based platform for deploying and managing permissioned Ethereum networks. It includes a Blockchain Platform Manager for provisioning and operating instances, and a Besu data plane for running the services and node roles required by a private Ethereum network.

Key Capabilities

Oracle Blockchain Platform Enterprise Edition 26.3 combines platform management, Besu network operations, secure application integration, wallet services, and digital-asset enablement in one Kubernetes-based deployment.

- **Lifecycle management:** Provision, configure, scale, upgrade, and monitor Besu instances through Blockchain Platform Manager and Kubernetes operators.
- **Permissioned Besu networks:** Operate QBFT validator networks with boot nodes for peer discovery, and RPC or archive nodes for specialized access and historical queries.
- **Secure blockchain integration:** Use the RPC Proxy as the governed application entry point for JSON-RPC, transaction orchestration, contract queries, wallet-backed signing, events, callbacks, and operational status.
- **Operational administration:** Use the Besu Service Console for node health, logs, block and transaction exploration, contract operations, and wallet-related administration.
- **Contract and digital-asset support:** Execute Solidity contracts in the EVM, maintain contract metadata, and support digital-asset contracts, SDKs, and sample applications for Besu.
- **Rich History and observability:** Optionally index blockchain transactions, events, traces, and state in Oracle Database or MySQL.
- **Enterprise security:** Integrate with OpenLDAP or an external OIDC provider; enforce RBAC; secure service communication.

Universal Wallet as a Service

Universal Wallet as a Service (UWaaS) provides custodial wallet and policy services for the Besu data plane. In this architecture, the UWaaS Wallet Manager is installed as a platform-level service and administered through the Platform Manager experience. The instance-level wallet service, also called the Local Wallet Service or OBP Besu Adapter, performs Besu-specific wallet, signing, reporting, and broadcast operations.

The RPC Proxy is the governed entry point for end-user token and wallet transactions, including signed and unsigned flows. It resolves caller and organization context, applies platform access controls, and coordinates with UWaaS and the local wallet service when signing or policy enforcement is required.

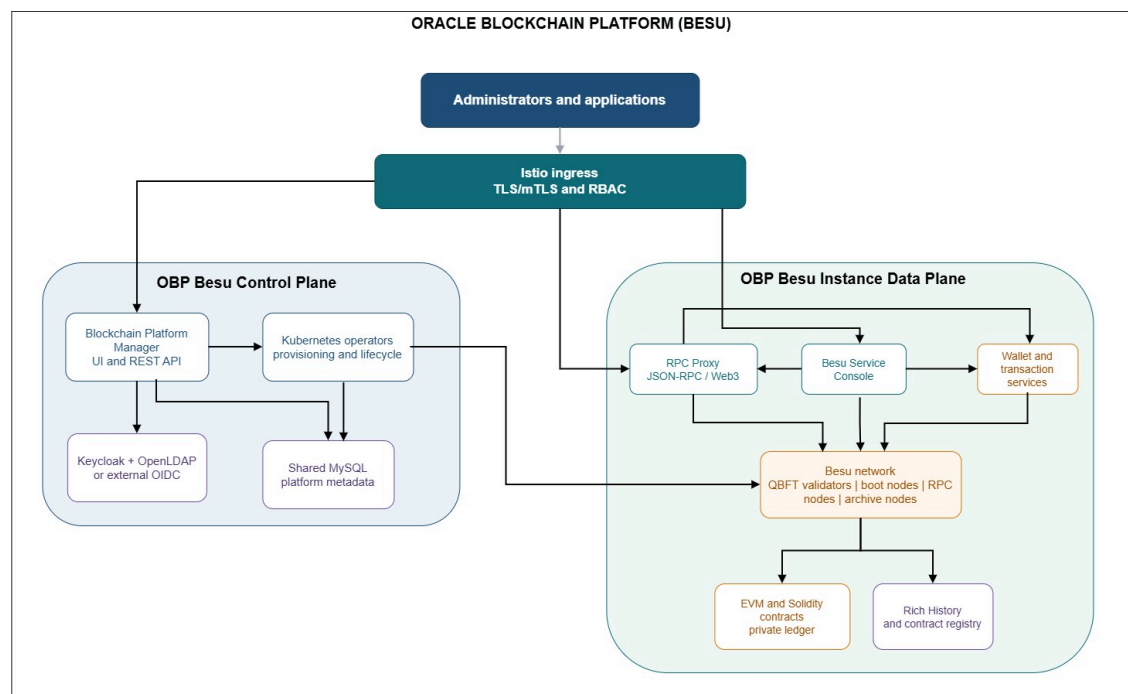
- **UWaaS Wallet Manager:** Provides platform-level wallet administration, organization and user wallet workflows, policy coordination, and rules-engine integration.

- **Local Wallet Service or Oracle Blockchain Platform Besu Adapter:** Provides instance-level wallet operations, signing, transaction broadcast, and reporting for a specific Besu network.
- **Policy and rules services:** Enforce granular roles and policy decisions for activities such as wallet administration, smart-contract deployment, and transaction execution.
- **Key management:** Integrates with protected secret or vault services for wallet encryption and key protection. Applications should not manage private keys directly.
- **Administration and RBAC:** Exposes UWaaS administration through the Platform Manager experience and controls it through platform roles. Wallet users do not require access to the UWaaS administration pages.

Architecture

Oracle Blockchain Platform Enterprise Edition 26.3 separates platform management from blockchain execution. The Blockchain Platform Manager provisions and manages Besu instances. Each Besu instance contains the services and node roles required to operate a private Ethereum network.

Architecture Diagram



Architecture Components

Architectural area	Components	Purpose
Kubernetes foundation	OKE/Kubernetes, namespaces, ingress, load balancers	Hosts and isolates control-plane and Besu instance components.
Control plane	Blockchain Platform Manager UI/API, Kubernetes operators, lifecycle APIs	Provisions, configures, scales, upgrades, and monitors Besu instances.

Architectural area	Components	Purpose
Identity and authorization	Keycloak, OpenLDAP, or external OIDC provider	Provides authentication, group-based RBAC, and user administration.
Security and networking	Istio service mesh, TLS/mTLS, cert-manager, Kubernetes Secrets	Secures ingress and internal service communication and manages certificates.
Besu access layer	RPC Proxy	Provides authenticated Ethereum JSON-RPC/Web3 access, transaction submission, queries, events, callbacks, and endpoint routing.
Operations interface	Besu Service Console	Provides node health, logs, block and transaction exploration, wallet operations, and smart-contract operations.
Wallet and transactions	UWaaS Wallet Manager, local wallet service, signing and transaction services	Manages wallets, signing, transaction submission, policy enforcement, and transaction status.
Besu network	Validator, boot, RPC, and optional archive nodes	Validators run QBFT consensus; boot nodes provide discovery; RPC nodes serve API traffic; archive nodes support historical queries.
Smart contract layer	EVM, Solidity contracts, contract registry	Executes smart contracts and stores contract metadata such as ABI and deployment information.
Rich History	Rich History plugin, Oracle Database or MySQL, archive node	Replicates and indexes blockchain transactions, events, traces, and state for SQL-based access.
Platform metadata	Shared MySQL with per-instance databases and users	Stores control-plane and instance metadata with logical isolation between instances.

Runtime Workflow

- Administrators authenticate through the configured identity provider and use Blockchain Platform Manager to create and manage instances.
- Kubernetes operators reconcile instance specifications and lifecycle requests into the required Besu services and node resources.
- Applications and Web3 clients access the blockchain through the RPC Proxy. Direct external access to validator and administrative Besu endpoints is not required.
- The RPC Proxy routes read and transaction requests to appropriate Besu nodes and can invoke wallet services when signing or policy enforcement is required.
- Besu validators use QBFT consensus to agree on blocks. Boot nodes support peer discovery, while RPC and archive nodes provide specialized access patterns.
- Optional Rich History components consume ledger information and expose indexed blockchain data through Oracle Database or MySQL.

Security and Deployment

- **Service security:** Istio provides ingress routing, authorization policy, and mTLS for internal platform services.
- **Certificate management:** Certificates are managed by cert-manager, with secrets scoped to the relevant Kubernetes namespace.
- **Peer communication:** Besu peer-to-peer communication uses the RLPx protocol. The RPC Proxy remains the governed entry point for application-facing blockchain API traffic.
- **Metadata isolation:** The control plane uses shared platform metadata storage while allocating logical database and credential isolation per instance.
- **Optional components:** Archive nodes and Rich History storage are optional and should be enabled when historical transaction, event, trace, or state queries are required.

Deployment Model

A Besu network consists of multiple Besu nodes connected via peer-to-peer (P2P) networking. Nodes run transactions in the Ethereum Virtual Machine (EVM), persist the blockchain ledger and state, and expose client access through RPC endpoints.

A Besu network includes the following node types.

- **Validator nodes:** Consensus-participating nodes that propose and validate blocks. Validator nodes must stay current with the canonical chain state to sign and vote correctly.
- **Boot nodes:** Nodes that support peer discovery so that other nodes can find and join the network.
- **RPC nodes:** Nodes that serve application traffic via JSON-RPC.
- **Archive nodes:** Nodes that retain all historical blockchain data and state (no pruning) to support deep queries, audits, and analytics.

Founder/Participant Model

Oracle Blockchain Platform Enterprise Edition for Besu supports two instance types: founder and participant.

A founder instance has the following capabilities.

- Creates and starts the private, permissioned Besu network.
- Operates in a single organization with independent authentication and administrative controls provided by Oracle Blockchain Platform Enterprise Edition.
- Establishes the initial network configuration and provides the baseline network endpoints used to add participants.

Other organizations can create participant instances and join the founder's network. The participant instance contains RPC nodes that connect to and join the founder's Besu network, so that the participant organization can submit transactions and query ledger data using standard Ethereum and RPC APIs. The participant instance can also operate in its own Oracle Blockchain Platform Enterprise Edition environment, and can be located in the same OCI tenancy or in a different OCI tenancy. Participant instances use RPC nodes to interact with the founder instance.

The following steps describe the basic operations you complete to work in the founder/participant model.

1. Create the founder network. The founding organization creates a founder Besu instance in Oracle Blockchain Platform Enterprise Edition, which initializes the private permissioned network.
2. Export genesis and node information from the founder. In the instance list in Blockchain Platform Manager, use the **Actions** menu to export the genesis and node information in .zip format.
3. Create a participant instance. The participating organization creates a participant Besu instance in Oracle Blockchain Platform Enterprise Edition by using the exported .zip file.
4. Check connectivity and synchronization between the instances. After both instances are running and network connectivity is in place between nodes, the RPC nodes connect and synchronize the ledger. After synchronization, transactions and blockchain data are visible in the participant's service console.
5. Independently administer the instances. Both organizations submit transactions to the same underlying ledger and interact with the same smart contracts, but each retains independent administrative control, which is enforced through the authentication and authorization implemented in their respective blockchain platform manager environments.

Required Infrastructure and Resources

This topic describes the infrastructure requirements, resources, and components needed for an instance.

Supported Compute Shapes

The following compute shapes are supported for Oracle Blockchain Platform Enterprise Edition for Besu:

Compute Shape

VM.Standard.E3.Flex
VM.Standard.E4.Flex
VM.Standard.E5.Flex

For more information on Flex Shapes, see Flexible Shapes.

Resource Estimates

The following table provides details on the minimum service and resource configuration used by Oracle Blockchain Platform Enterprise Edition for Besu. Ensure that these are available in your tenancy or the stack creation will fail.

Service or Resource	Estimated Base Usage
Oracle Kubernetes Engine (OKE)	1 OKE cluster
OKE Node Pool	1 Note: If you're manually creating the stack, the OKE node pool must be set to 39 pods per node. If a new node pool is being created for additional instances, ensure it is set to 14 pods per node minimum.
Container Registry (OCIR)	20 This is used to store container images for Oracle Blockchain Platform Enterprise Edition components. Ensure 25GB of space is available.

Service or Resource	Estimated Base Usage
Virtual Cloud Network (VCN)	1 VCN, used by the cluster
Load Balancer and IP Addresses	2 or more, 1 for Blockchain Platform Manager, 1 for each Besu instance

Cluster Requirement for Besu

Each instance requires a minimum of 1 worker node.

Number of Instances	Worker Node Count	Worker Node Configuration	Worker Node Boot Volume	Load Balancer/IP Address Count
1	3	4 OCPU 64GB memory	150GB	2 total 1 Blockchain Platform Manager 1 instance
2	3	4 OCPU 64GB memory	150GB	3 total 1 Blockchain Platform Manager 2 instances

Instance Components

A deployed instance of the Besu network provides the following node components.

Component	Default	Minimum	Maximum	Description
Validator nodes	4	4	7	Besu nodes that propose, validate, and add blocks of transactions to the ledger. They use the Quorum Byzantine Fault Tolerance (QBFT) consensus protocol to agree on each block before it is added.
Boot nodes	2	1	2	Boot nodes act as initial connection points, so that new or restarting nodes can discover the network.
RPC nodes	2	1	3	Besu nodes that expose the Ethereum JSON-RPC (JavaScript Object Notation Remote Procedure Call) APIs. They let applications and tools read blockchain data and submit transactions without running their own node.
Archive nodes				Archive nodes are used by rich history and analytics.

The instance includes managed components such as an RPC Proxy, a console, wallet storage, and related infrastructure, providing a complete environment to perform lifecycle management (LCM) operations, manage the instance's network, and submit transactions through the RPC Proxy.

Component	Description
RPC proxies	Oracle services that are used to authenticate, manage, and scale access to Web3 JSON-RPC APIs.

Component	Description
Service consoles	Oracle services that let you administer a Besu network through a web console or APIs. A user's access depends on their assigned role and privileges.
Wallet service	An Oracle custodial wallet service that can register and manage user key pairs, sign user transactions, and send transactions to the Besu network for processing.

Getting Started with Oracle Blockchain Platform Enterprise Edition for Besu

The following is a basic task flow you can follow when new to Oracle Blockchain Platform Enterprise Edition for Besu.

Table 1-1 Basic Workflow

Summary	Role	Description	Documentation
Deploy Oracle Blockchain Platform Enterprise Edition	Administrator	The administrator deploys Oracle Blockchain Platform Enterprise Edition from OCI Marketplace into a supported OCI environment and completes the prerequisites, including an external MySQL or Oracle Database and one supported identity provider.	Deploy from Oracle Cloud Marketplace
User Management	Administrator	An administrator logs in to the Control Plane, configures the identity provider (LDAP or OIDC/IDCS), and maps users and groups to the appropriate RBAC roles.	Log into Blockchain Platform Manager for the First Time User Management with OpenLDAP
Provision a Network	Administrator	The administrator provisions a Besu instance/network, including the organization and access settings, and verifies that it is healthy and operational through the service console.	Manage the Organization and Network
Configure Wallets	Administrator	The administrator configures the wallets/accounts and endpoint access required by the application (RPC Proxy and UWaaS, where applicable).	Create an Organization and Wallet

Table 1-1 (Cont.) Basic Workflow

Summary	Role	Description	Documentation
Develop Smart Contracts	Developer	A developer uses Blockchain App Builder to create and deploy a custom smart contract or registers/reuses an existing contract via RPC Proxy. The flow must correctly resolve founder/participant organization IDs and support the no-op or governance path, as applicable.	Using the Blockchain App Builder Extension for Visual Studio Code
Work with Smart Contracts	User and Business User	The user can interact with the smart contract from Postman or an existing application via the supported RPC Proxy APIs, including submitting and querying transactions and retrieving the required history. A business user completes the intended transaction end to end.	Web3 Transactions and Besu Instance APIs
Administer Smart Contracts	Administrator	An administrator can monitor health, status, and logs, manage access, and recover from documented failures.	

Additionally, two bundled digital assets applications are packaged with Oracle Blockchain Platform Enterprise Edition. You can deploy these applications by using the one-click application deployment workflow. See [One-Click Application Deployment](#). For more information about the applications, see Wholesale Central Bank Digital Currency Application and Deposit Token Application in the Digital Assets Guide.

2

Deploy Oracle Blockchain Platform Enterprise Edition for Besu

The following steps are a high-level overview of how to deploy Oracle Blockchain Platform Enterprise Edition for Besu.

Provision Blockchain Platform

Use Oracle Cloud Marketplace to provision Blockchain Platform.

1. Enter information for your Blockchain Platform, OCIR image, and Kubernetes cluster.
2. Log in to Blockchain Platform Manager.

See [Deploy from Oracle Cloud Marketplace](#).

Configure Your Identity Provider

You can choose to use the built-in OpenLDAP service or use your own identity provider. You can then add users and groups needed by various components. See [User Management and Identity Providers](#).

Rich History Database

You can optionally index blockchain transactions, events, traces, and state in Oracle Database or MySQL. See [Create a Rich History Database Configuration](#).

Configure a Wallet

The Instance or Local Wallet Service provides APIs for querying and transacting on the instance network by using the signing, transaction, query, and broadcast capabilities of the service.

1. Log in to the Blockchain Platform Manager using the administrator credentials that you provided during installation.
2. On the **Wallet** tab, configure the wallets and accounts required by your application.

See [Create an Organization and Wallet](#).

Create an Instance

Use Blockchain Platform Manager to set up your authentication server, create an administrative user, and create your instance.

1. Log in to the Blockchain Platform Manager using the administrator credentials that you provided during installation.
2. Set the default identity provider (either OpenLDAP or IDCS).
3. Add a user with permissions to create instances.
4. Log in as the new admin user and create an instance.

For complete steps, see [Create an Instance](#).

Provision a Besu Network

After you create an instance, you can open the Oracle Blockchain Platform service console to work with Besu nodes and to explore blocks in the ledger. You can also deploy contracts, create wallets, and send transactions by using the RPC proxy as described in [RPC API for Oracle Blockchain Platform Enterprise Edition for Besu](#).

For more information, see [Manage the Organization and Network](#)

Deployment Prerequisites

In order to deploy Oracle Blockchain Platform Enterprise Edition, you must have the following:

General Prerequisites

- A basic understanding of blockchain concepts and Besu. For more information about Besu, see [Besu Ethereum client](#).
- An Oracle Cloud account:
 - Sign up for an Oracle Cloud account: [How Do I Sign Up?](#)
 - Sign in for the first time: [Sign In to Oracle Cloud For the First Time](#)
 - Sign in to Oracle Cloud: [Oracle Cloud Account](#)

OCI Prerequisites

- Access to an assigned Oracle Cloud tenancy with permissions to create Oracle Kubernetes Engine (OKE) nodes. You must create a compartment in Oracle Cloud Infrastructure in your assigned tenancy. See:
 - [Setting Up Your Tenancy](#)
 - [Managing Compartments](#)
- OCI Registry credentials (user name and auth token) with permissions to push container images to the repository.
- Generate an OCIR Auth Token. You can generate this in your OCI My Profile under Auth Tokens, Generate Token. See [Getting an Auth Token](#). To generate a token:
 1. In the OCI console, select **Profile** and then select your user account.
 2. Open **Tokens and keys**.
 3. Under **Auth tokens**, create a token and then copy it and store it securely.
 4. Determine your OCIR user name, which typically uses the following format: <tenancy-name>/oracleidentitycloudservice/<username>. For example, acmeinc/oracleidentitycloudservice/dev.user@example.com.

Rich History Database Prerequisites

The rich history database is external to Oracle Blockchain Platform and contains data about the blockchain ledger's transactions on the channels you select. You use this database to create analytics reports and visualization about your ledger's activities.

- You must have a MySQL or Oracle Database provisioned and configured for the rich history database.
- You must ensure the database port is open, and add an ingress rule that enables the rich history database to access the database.

Identity Provider Prerequisites

By default, an OpenLDAP server is included and configured for testing purposes.

If you want to use an external identity provider, you need to create a confidential application as described here: [User Management with an External Identity Provider](#).

Deploy from Oracle Cloud Marketplace

As an Oracle Cloud Infrastructure administrator, you can install Oracle Blockchain Platform Enterprise Edition for Besu from Oracle Marketplace.

To find Oracle Blockchain Platform Enterprise Edition in Oracle Cloud Marketplace:

1. Log in to [Oracle Cloud Marketplace](#).
2. From the Oracle Cloud Marketplace home page, use the search box under All Applications and search for the keywords Oracle Blockchain Platform.
3. From the search results, select Oracle Blockchain Platform Besu Enterprise Edition.

After finding the Oracle Blockchain Platform Enterprise Edition listing in Oracle Cloud Marketplace, you can deploy Oracle Blockchain Platform using the provided stack listing. The stack prompts you for specific information and then builds an Oracle Kubernetes Engine Cluster with Oracle Blockchain Enterprise Edition installed.

1. From the Marketplace page, select Oracle Blockchain Platform Besu Enterprise Edition.
2. Select **Launch Stack**.
3. On the Launch Stack page, provide the following information:

- a. Select **Compartment**

Specifies the compartment where the resources will be built.

- b. Select **Version**

Provides a list of versions that are available. Select the version of Oracle Blockchain Platform Enterprise Edition that you want to provision: 26.3.1.

- c. **Terms and Conditions**

Review the licenses before proceeding with the instance creation.

Select **Launch Stack**.

4. Optionally, customize the **Stack Information** fields. All fields are completed with defaults.
 - a. **Custom provider**
Custom providers are not supported at this time.
 - b. **Name**
Name of the stack. It has a default name and provides a date time stamp. You can edit this if desired.
 - c. **Description**
Description of the stack that you are creating.
 - d. **Create in compartment**
Defaults to the compartment you selected on the Oracle Blockchain Platform Marketplace page.

- e. Terraform version
Ensure version 1.5.x or later is selected.
- f. Tags
Optional. Tags are a convenient way to assign a tracking mechanism.

Click **Next**.

5. Fill in the required details to configure the platform on the **Configure Variables** page.

Blockchain Platform Configurations

- a. Domain Name
The domain name that will be used for your Oracle Blockchain Platform Manager. This will be used in the Platform Manager URL as described in [step 7](#).
- b. Blockchain Platform Manager Admin User Password
This is used to set the Blockchain Platform Manager admin user's password.
- c. LDAP User Password
This is used to set the admin user's password for the LDAP authentication server.

OCIR Image Configurations

- a. OCIR Username
User name used to log into Oracle Cloud Infrastructure Registry. Enter the name of the user in the format:

<tenancy-namespace>/<username>

where <tenancy-namespace> is the autogenerated Object Storage namespace string of the tenancy in which to create repositories (as shown on the **Tenancy Information** page). For example

ansh81vru1zp/jdoe@example.com

Note that for some older tenancies, the namespace string might be the same as the tenancy name in all lowercase letters. For example,

example-dev

If your tenancy is federated with Oracle Identity Cloud Service, use the format

<tenancy-namespace>/oracleidentitycloudservice/<username>

See [Logging in to Oracle Cloud Infrastructure Registry](#).

- b. OCIR Auth Token
The auth token used to access OCIR. This prerequisite is described here: [OCI Prerequisites](#).

Kubernetes Cluster Configurations

- a. Cluster Name
Name of the OCI Kubernetes Engine cluster that will be created.

- b. Node Pool Name
Name of the node pool for the worker nodes.
 - c. Enable Cluster Autoscaler for Node Pool
This will enable node pools to autoscale based on resource usage and will add and remove worker nodes as needed.
 - d. Minimum Number of Worker Nodes
The minimum number of nodes in the node pool. If autoscaling has not been enabled, this is the total number of worker nodes available. At a minimum, three worker nodes are required.
 - e. Maximum Number of Worker Nodes
The maximum number of nodes in the node pool. This is only available if autoscaling has been enabled.
 - f. Worker Nodes Instance Shape
Select the appropriate compute shape. For information on supported shapes and their configurations, see: [Supported Compute Shapes](#)
 - g. Worker Node OCPU Count
The number of OCPUs for each worker node compute instance. The minimum is 4 OCPUs.
 - h. Worker Node Memory (GB)
The amount of memory for each worker node compute instance. The minimum is 64GB.
6. On the **Review** page, review the information you provided and click **Create**.
 - **Run Apply** is selected by default. It will run the stack and set up your Blockchain Platform Manager.
 - If **Run Apply** was not selected, click **Apply** and select **Automatically Approve for Apply job plan resolution**.

It will take approximately 30 minutes to complete the installation.
7. After the job has completed, a new tab is available: **Application Information**.
 - a. Copy the **Domain Hosts Entry** field and append it to your client machine's hosts file:
 - macOS or Linux: /etc/hosts
 - Windows: C:\Windows\System32\drivers\etc\hosts
 - b. Optional: Copy the **Kubernetes Configuration** field and append it to your client machine's kubeconfig file (~/.kube/config) to access the cluster.
8. You can now go to the Blockchain Platform Manager at the following URL:
`https://controlplane.<domain_name>/console/index.html`
Where <domain_name> corresponds to the domain name that was submitted in the Blockchain Platform Configurations details in [step 5](#).
For example: `https://controlplane.obpee.com/console/index.html`

3

Blockchain Platform Manager

After you install Oracle Blockchain Platform Enterprise Edition for Besu, you can access Blockchain Platform Manager to create, configure and manage platform instances.

Log into Blockchain Platform Manager for the First Time

Before you begin, review the **Application Information** page, which shows details about the Blockchain Platform Manager, Kubernetes cluster, and Terraform output.

- Note the ingress gateway IP address, which is used for Blockchain Platform Manager, and the ingress IP address for instance services, which is used by the service console and RPC gateway.
- Full job logs, including jump host details and Terraform logs, are available under **Jobs**.

To ensure proper host name resolution, add entries to your local `etc/hosts` or `C:\Windows\System32\drivers\etc\hosts` file for Blockchain Platform Manager and OpenLDAP fully-qualified domain names. For a production environment, use DNS and certificates instead of `etc/hosts` file mappings. Add entries similar to the following example, where the the ingress IP address is the ingress gateway IP address shown in the installation output.

```
<ingress-ip> controlplane.<domainname> openldap.<domainname> auth.<domainname>
```

Log into Blockchain Platform Manager for the first time:

1. Open Blockchain Platform Manager at `https://controlplane.domainname/console/index.html`. Your browser might warn you about a self-signed certificate.
2. Log in to Blockchain Platform Manager by using the `obpadmin` user name and the password that you provided during installation.
3. If you are using an external identity provider, proceed with the instructions in [User Management with an External Identity Provider](#) to create a confidential application, enable the identity provider, and add your users.
4. If you are using the built-in default OpenLDAP authentication server:
 - On the **Configuration** page, select **Authentication Servers**.
 - Select the Default authentication server, then select **Save and Set active**.
 - Add an instance administrative user to the system. Do not use the Oracle Blockchain Platform administrator account for routine tasks. Log out and then log back in as the new user.
 - On the **Authentication Servers** tab of the **Configuration** page of Blockchain Platform Manager, click **Add User**.
 - Once you've entered the user name and password, this user will be added to the LDAP server as an administrative user.
 - You can now log out of Blockchain Platform Manager with your default admin user, and log in with this newly added instance administrative user to create an instance.

Instances Page Overview

Use the Instances page to create and manage blockchain platform instances (nodes and networks). Select **Create Instance** to create an instance. For more information, see [Create an Instance](#).

This page lists all instances and the following instance details.

- **Role:** Founder instances were created as a separate network. Participant instances were created to add nodes to an existing network.
- **Shape:** The resource configuration of the instance. The Enterprise shape is the only supported shape.
- **State:** Creating, Up, Stopped, and so on.
- **Domain Name:** The domain name that you specified when creating the instance.
- **Created:** Instance creation date/time.
- **Version:** Version number of the instance.
- **Remarks:** Additional remarks to indicate instance state.

The **Actions** menu for each instance lets you:

- Start, stop, and terminate instances.
- Select **Confidential Client Credentials** to generate or refresh access tokens.
- Select **Instance URLs** to see the connection URLs for services provided by the Besu instance.
- Select **Deploy Applications** to deploy one of the sample applications.
- Select **Export Genesis Metadata** to generate a .zip file that participant network members can use to join a network.

Wallet Page Overview

Use the Wallet page to track and manage all organizations, wallets, and policies.

Activities Page Overview

All lifecycle management actions completed on an instance create a traceable activity record. The activity record logs all lifecycle phase changes the instance goes through from creation to termination. You can filter the activity log by instance, time range, or operation.

This page lists all activities and the following details.

- **Instance Name**
- **Operation:** Start, Stop, Create, Terminate, and so on.
- **Operation Status:** The status of the last action on any instance.
- **Start Time**
- **Last Updated Time**
- **Initiated By:** The user who initiated the activity.

Configuration Page Overview

Use the Configuration page to configure authentication servers and Blockchain Platform Manager itself.

Select **Authentication Servers** to configure authentication and users. For more information, see [User Management with OpenLDAP](#).

Select **Rich History Database** to configure the database for the rich history.

Select **Platform Manager Settings** to view or edit the platform manager name and timeout values.

Deployed Samples Page Overview

This page shows details of the sample applications you've deployed.

4

User Management and Identity Providers

You can choose to use the built-in OpenLDAP service or use your own identity provider.

In addition to LDAP-based authentication, Blockchain Platform Manager supports integration with external Identity Management systems (IdMs) via OpenID Connect (OIDC). This enables the use of industry-standard authentication protocols for enhanced interoperability.

If you use OpenLDAP, you can create users in LDAP which can then log into the Blockchain Platform Manager.

If you use an external OIDC provider, you must create a confidential client in the provider, log in to Blockchain Platform Manager, save the configuration with the client credentials of the confidential client, and then save and mark it as active. You will then get a Blockchain Platform Manager callback URL which you can configure as a redirect URI in the confidential client in the provider. After this, any user in the provider assigned to confidential client can log in to the Blockchain Platform Manager console.

All groups must be created in the provider as expected in Blockchain Platform Manager, desired users must be mapped to the groups and both the users and the groups must be assigned to confidential client in the provider. Role-based access control in the platform enforced through group membership of users.

Note

Only one identity provider is allowed per installation. You cannot enable both the default OpenLDAP identity provider and then enable an external OIDC provider.

User Management with OpenLDAP

Blockchain Platform Manager uses an integrated OpenLDAP server for initial Identity and Access Management (IAM) during installation. This OpenLDAP server manages user credentials and enforces Role-Based Access Control (RBAC) using preconfigured groups. All user information such as credentials and group memberships is stored on this OpenLDAP server.

Enabling the Default LDAP Server

If you choose to enable the default LDAP server for testing, do the following:

1. Open Blockchain Platform Manager at <https://controlplane.domainname/console/index.html>.
2. Log in to Blockchain Platform Manager by using the `obpadmin` username and the password that you provided during installation.
3. On the **Configuration** page, select **Authentication Servers**.
4. Select **Save and Set Active**.
5. Add a new user by selecting **Add User**.

6. Create the Blockchain Platform Manager Administrator user. Once you log out of Blockchain Platform Manager you can log back in with this user rather than the default obpadmin user created by the installation.
7. Add any other users as needed. See [User Groups and Roles](#) for details about authorization.

Token Issuance and Group Membership Propagation

When an instance is created, Blockchain Platform Manager configures the authentication server to enable token issuance with required claims, including user identity and relevant client/party information. Each token includes group membership information, encapsulated in a payload claim. Instance components use these claims to authorize or block external access to workload pods.

You can add users directly to the OpenLDAP server by using OpenLDAP browsers such as jXplorer. Blockchain Platform Manager administrators can use the administrator user name and password that was provided during installation to connect to `openldap.<cp-name>.<cp-domain>:443` with SSL enabled. Once connected, administrators can then add users and assign or modify groups to give the appropriate access levels.

User Management with an External Identity Provider

This topic describes how to configure IDCS as an external OpenID Connect (OIDC) identity provider for Oracle Blockchain Platform Enterprise Edition.

Prerequisites

Before creating an instance in Oracle Blockchain Platform Enterprise Edition:

- Create an administrator account in the selected identity provider.
- Create users and groups required for Oracle Blockchain Platform Enterprise Edition administration and blockchain access.
- Register a confidential OIDC client application.
- Obtain the client ID, client secret, issuer, authorization, token, JWKS, and revocation endpoints.
- Determine the claim names used for the username, client ID, and group memberships.

Configure IDCS

Configure the confidential client

1. Sign in to the IDCS identity domain. Create a domain.
2. Create a confidential client application for the Oracle Blockchain Platform Enterprise Edition control plane.
 - a. Navigate to **Profile**, and then **Identity Domain**, and then **Integrated applications**.
 - b. Select **Add application button**, and then **Confidential Application**, and then **Launch Workflow** and provide a name before submitting to create the confidential application.
 - c. After the application is created, select the application and navigate to the OAuth configuration tab. Select **Edit OAuth configuration**.
 - d. Select these options for Resource Server Configuration:
 - Configure this application as a resource server now

- Allow token refresh
- Primary Audience: obpee-cp
- e. Select these options for Client Configuration:
 - Configure this application as a client now
 - For **Allowed grant types**, select
 - Resource owner
 - Refresh token
 - Authorization code
 - Client type: Confidential
- f. Certificate:
 - Select **Bypass consent**
 - Client IP address: Anywhere
- g. Token issuance policy:
 - Authorized resources: All
- 3. Create groups and users in the domain. Navigate to **Profile**, and then **Identity Domain**, and then **User management**. See [User Groups and Roles](#) for details about the groups and how they're used.
- 4. Configure the required redirect URI for the Oracle Blockchain Platform Enterprise Edition control plane and service console.
- 5. Assign the users and groups to the confidential client application. Navigate to **Confidential Application**, and then **Users** and **Confidential Application**, and then **Groups** to do this.

Configure group claims

To include group memberships in access tokens:

1. Open the OAuth configuration for the confidential client.
2. Add the **Identity Domain Administrator** application role.
3. Use an administrator account to obtain an access token through the token endpoint. The administrator must belong to the IDCS_Admistrators and, where applicable, OCI_Admistrators groups.
4. Use the access token to create a custom access-token claim.

```
curl -sS -X POST "https://<identity-domain-host>/admin/v1/CustomClaims" \
-H "Authorization: Bearer $ACCESS_TOKEN" \
-H "Content-Type: application/json" \
--data-binary '{
  "schemas": ["urn:ietf:params:scim:schemas:oracle:idcs:CustomClaim"],
  "name": "group_roles",
  "value": "$user.groups.*.display",
  "expression": true,
  "mode": "always",
  "tokenType": "AT",
  "allScopes": true
}'
```

Use `group_roles` as the **Groups Claim Name** in the Oracle Blockchain Platform Enterprise Edition configuration.

Configure Oracle Blockchain Platform Enterprise Edition

Enter the identity provider values in the Oracle Blockchain Platform Enterprise Edition external OIDC provider configuration.

1. Open Blockchain Platform Manager at `https://controlplane.domainname/console/index.html`. Your browser might warn you about a self-signed certificate.
2. Log in to Blockchain Platform Manager by using the `obpadmin` username and the password that you provided during installation.
3. On the **Configuration** page, select **Authentication Servers**.
4. Select **Add Identity Provider**
5. Complete the following configuration fields on the Add Identity Provider page:

Field	Required	Description
Well-known configuration URI	Yes	OIDC discovery URL for the identity provider. This is Domain URL/.well-known/openid-configuration The Domain URL can be found in Identity and Security , and then Identity , and then Domains , and then Domain URL
Issuer URL	No	Issuer used to validate token origin.
JWKS URL	No	Endpoint used to validate token signatures.
Authorization endpoint	No	Endpoint used for browser-based authentication.
Token endpoint	No	Endpoint used to exchange the authorization code for tokens.
Revocation endpoint	No	Endpoint used to revoke refresh tokens.
Scope	Yes	OIDC scopes, for example openid email profile offline_access.
Client claim name	Yes	Claim containing the client identifier, such as cid or azp.
Client claim value	Yes	Expected value of the client identifier claim.
Client ID	Yes	Confidential client identifier.
Client secret	Yes	Confidential client secret.
User claim name	Yes	User identifier claim, such as preferred_username or email.
Blockchain Platform Manager (BPM) Admins	Yes	Provider group mapped to Blockchain Platform Manager administrators.
Groups claim name	Yes	Claim containing group memberships, such as groups or group_roles.

6. Complete the group name fields as required. These can be found here: [Groups](#)
7. Once all fields are populated, click **Save and Set Active**.

Ensure Domain Settings have Configure Client Access Enabled

Ensure that the domain settings have Configure Client Access enabled. To verify or enable this:

1. Access the IDCS console at:

`<Domain-URL>/ui/v1/adminconsole`

2. Sign in with a user that has the Identity Domain Administrator role (or equivalent tenant/domain admin permission).
3. Navigate to **Settings tab**, and then **Domain settings - Access signing certificate** and ensure that **Configure client access** is enabled.
4. If it is disabled, enable it using **Edit domain settings**.

Add a Redirect URL for Blockchain Platform Manager

After logging out of Blockchain Platform Manager, when you attempt to log back in it will go through your new identity provider. This requires that you add a redirect URL.

For example, if you are using IDCS, complete the following steps:

1. Log in to OCI and go to Identity and Security.
2. Select **Integrated Applications**.
3. Select the confidential application you created.
4. Select **OAuth Configuration**, and click **Edit OAuth Configuration**.
5. Add a redirect URL to the list with the URL for your Blockchain Platform Manager URL. Click **Submit**.

Note

- After you create your instance, you will need to add another redirect for your Blockchain Platform console URL. It is instance-specific and cannot be added until after the instance is created.
- If you deploy a sample application, you will need to add another redirect for the sample application URL.

User Groups and Roles

This overview describes the groups and roles that are relevant to Oracle Blockchain Platform. Anyone who uses or administers Oracle Blockchain Platform must be added to the authentication server and granted the correct group.

Groups

Global role groups are used with both OpenLDAP and OIDC.

For OpenLDAP, Blockchain Platform Manager creates the global groups. Their names are fixed constants and are the same for all instances.

For OIDC, you must create and configure the global groups in the external identity provider. The group names do not need to follow a fixed naming pattern.

For instance-specific access, you must explicitly create the instance access group in OpenLDAP and OIDC if you do not use the default access group, Instance Admin.

Below are the group roles that are available for Oracle Blockchain Platform.

User Group	Group Name in OpenLDAP	Description
BPM Admin	OBP_CP_ADMIN	Platform-level administrator users with membership to this group can login in to Blockchain Platform Manager and set/reset/configure IdP providers, rich history, KeyStore, add users, assign roles in OpenLDAP. They cannot create/delete/update/view instances.
Instance Admin	OBP_INSTANCE_ADMIN	Main administrative group for Besu instance operations: create, delete, update, start, stop, scale, certificate updates. This group can create/delete/scale/stop/start or perform any other life cycle operation on the instances. They can also access service console and perform all the operations on it via UI or console REST APIs (if present). These users have read/write access to both the Blockchain Platform Manager and Blockchain Platform console.
Instance Operator	OBP_INSTANCE_OPERATOR	Read/view access: instance details, operation status, console/API viewing paths, but not lifecycle operations. This group can also access Blockchain Platform Manager and can view instance details (provided the user is part of the access group of the instance as well). The permissions will be read-only and they will not be able to perform any other operations.
Instance API Client	OBP_INSTANCE_API_CLIENT	Proxy client group. Intended for users/clients that interact through RPC proxy APIs. Not able to log in to the Blockchain Platform Manager or data plane. Only REST API access allowed.
Fabric CA Admin	OBP_FABRIC_CA_ADMIN	A Fabric-specific role used to manage Fabric CA identities, such as registering and enrolling users in CA. It does not provide Blockchain Platform Manager administration, instance management, console access, or Besu/RPC access. Not able to log in to Blockchain Platform Manager or Blockchain Platform Console. Specific to Fabric CA.
Wallet Super Admin	OBP_WALLET_SUPER_ADMIN	Allows creation of organization administrators and organization users. Has full access for configuring wallets and policies across multiple orgs. Can log in to Blockchain Platform Manager.

User Group	Group Name in OpenLDAP	Description
Wallet Org Admin	OBP_WALLET_ORG_ADMIN	Allows creation of org admins and org users in their own org. Has full access for configuring wallets and policies only in their org. Can login to Blockchain Platform Manager.
Wallet Org User	OBP_WALLET_ORG_USER	Has read access for all wallets and policies in their own organization. Cannot login to Blockchain Platform Manager.
Digital Assets Super Admin	OBP_DA_SUPER_ADMIN	Creates a new asset, assigns one or more deployer for the asset, assigns approver based on the GOV policy. Cannot login to Blockchain Platform Manager.
Digital Assets Token Admin	OBP_DA_TOKEN_ADMIN	Manages the lifecycle of the asset Cannot login to Blockchain Platform Manager.
Digital Assets Deployer	OBP_DA_DEPLOYER	Develop and initiate approval for for initial deployment or upgrades. Cannot login to Blockchain Platform Manager.
Digital Assets Approver	OBP_DA_APPROVER	Approves asset deployment. Cannot login to Blockchain Platform Manager.

Global Groups with Optional Instance-specific Overrides

Configuring global groups once at the identity-provider level would define role aspect for the user for any instance.

Additionally, an instance specific access group defining user entitled for operation on that instance needs to be defined. The users added to the group would have permissions to act or operate on that instance with the role defined by the global groups. If a user is not in the access group for the instance, he cannot perform any operation on the instance irrespective of roles defined by the global group for him.

During instance creation, the administrator will additionally provide a group name to be used as access group for the instance which defines the set of users who can operate on the instance (in different roles defined by the users association to the global roles group). This input is taken as a field during instance creation rather than using a group name with a predefined naming template (including the instance name), for the following reasons:

- Organizations may have their own naming conventions to define groups in their IDP and if the product pre-decided a name template to be used, then it may not be acceptable to you. Instead this approach gives you the control to define the group name for the instance access group.
- If you want the same set of users to be able to operate on multiple instances, you can create a group and re-use it for multiple instances.

This combines the unified global group's simplicity with the permissioning flexibility specific for instances.

Important points to consider:

- The access group field will be optional with a default value of instance admin's group, named OBP_INSTANCE_ADMIN.
- If no explicit access group is given and the default OBP_INSTANCE_ADMIN is used, then instance will lose any exclusive access pattern specific for this instance, because OBP_INSTANCE_ADMIN is a global group and any instance admin users from the IdP environment will get access to the instance.
- Access group chosen once, can't be modified later.
- Creating the access group and adding the required user, is user's responsibility both in OpenLDAP and OIDC. Blockchain Platform Manager will not create any groups automatically.
- Once a user is added in a group, a fresh login needs to be done in Blockchain Platform Manager. Blockchain Platform Manager will not recognise the user's access in an existing session created before group enrollment.

Besu Role Based Access Control

Table 4-1 Besu Role Based Access Control

Role Name	Log in to Blockchain Platform Manager	Create Instance	Act on an Instance	View Instance Details	Configure Authentication	View Wallet Tab	Perform Actions in the Wallet Tab	Call UWaas APIs	Access Service Console	Execute RPC Proxy APIs
BPM Administrator	Yes	No	No	No	Yes	No	No	No	No	No
Instance Administrator	Yes	Yes	Yes Only for organizations where they're a member of the access group	Yes Only for organizations where they're a member of the access group	No	No	No	No	Yes Only for organizations where they're a member of the access group	Yes Only for organizations where they're a member of the access group
Instance API Client	No	No	No	No	No	No	No	No	No	Yes Only for organizations where they're a member of the access group

Table 4-1 (Cont.) Besu Role Based Access Control

Role Name	Log in to Blockchain Platform Manager	Create Instance	Act on an Instance	View Instance Details	Configure Authentication	View Wallet Tab	Perform Actions in the Wallet Tab	Call UWAas APIs	Access Service Console	Execute RPC Proxy APIs
Instance Operator	No	No	No	Yes Only for organizations where they're a member of the access group	No	No	No	No	Yes Read-only access for only the organizations where they're a member of the access group	No
Wallet Super Administrator	Yes	No	No	No	No	Yes	Yes	Yes	Yes	No
Wallet Organization Administrator	Yes	No	No	No	No	Yes Only for their own organization	Yes Only for their own organization	Yes	Yes	No
Wallet Organization User	No	No	No	No	No	No	No	Yes Only GET calls	Yes	No

Service Console Views

RBAC group	Dashboard	Nodes	Explorer	Contracts	Digital Assets	Developer Tools	Accounts	Logs
BpmAdminGroup	No: Console access alone	No: Console access alone	No: Console access alone	No: Console access alone	No: Console access alone	No: Console access alone	No: Console access alone	No: Console access alone

RBAC group	Dashboard	Nodes	Explorer	Contracts	Digital Assets	Developer Tools	Accounts	Logs
InstanceAdminGroup	Visible: all dashboard summaries	Visible: node inventory and details; administrative node actions	Visible: blocks, transactions, and search	Visible: registry data, contract details, and contract operations	Visible: all Digital Assets views	Visible	Visible: current user's accounts, balances, and activity	Visible
InstanceOperatorGroup	Visible: read-only summaries	Visible: node inventory and details; no administrative node actions	Visible: blocks, transactions, and search	Visible: read-only	Visible: read-only	Visible	Not shown	Visible
InstanceApiClientGroup	No: Console access alone	No: Console access alone	No: Console access alone; enables Explorer when combined with DaToken AdminGroup	No: Console access alone	No: Console access alone	No: Console access alone	No: Console access alone	No: Console access alone
DaTokenAdminGroup	Not shown	Not shown	Shown only when combined with InstanceApiClientGroup	Visible: registry data, contract details, and permitted contract operations	Visible: Digital Assets views and permitted token-administration workflows	Not shown	Visible: current user's accounts, balances, and activity	Not shown
WalletOrgAdminGroup	Not shown	Not shown	Not shown	Not shown	Not shown	Not shown	Visible: current user's accounts, balances, and activity	Not shown
WalletOrgUserGroup	Not shown	Not shown	Not shown	Not shown	Not shown	Not shown	Visible: current user's accounts, balances, and activity	Not shown

RBAC group	Dashboa rd	Nodes	Explorer	Contract s	Digital Assets	Develop er Tools	Account s	Logs
WalletS uperAdm inGroup	Not shown	Not shown	Not shown	Not shown	Not shown	Not shown	Visible: current user's accounts, balances, and activity	Not shown

Combined-role behavior

- InstanceOperatorGroup plus any wallet group receives the Operator tabs plus Accounts. The wallet role does not grant contract operations or administrative node actions.
- DaTokenAdminGroup plus InstanceApiClientGroup receives Contracts, Digital Assets, Explorer, and Accounts.
- InstanceAdminGroup receives every current Console tab. Additional groups may grant access to downstream services but do not add Console tabs.
- BpmAdminGroup and InstanceApiClientGroup are not standalone Console login roles.
- Wallet organization and super-administrator group names do not currently grant cross-user wallet inventory or wallet mutation in the Console. The Accounts tab remains scoped to the authenticated user's wallets.

5

Create a Rich History Database Configuration

Rich history can index blockchain transactions, events, traces, and state in Oracle AI Database or MySQL. Administrators can use SQL queries for investigation, reporting, reconciliation, and integration with off-chain data. In consortium deployments, members can share a founder-provided rich history database or maintain separate databases for their own nodes.

If you want a rich history database associated with your instance, it must be configured before the instance is created. You must have created either an Oracle Database or MySQL database before configuring a rich history database in Blockchain Platform Manager.

1. Open Blockchain Platform Manager (<https://controlplane.domainname/console/index.html>).
2. Log in to Blockchain Platform Manager by using the newly created user name and the password that you added when creating your identity provider.
3. On the Configuration Page, select the Rich History Database tab.
4. Enter all the required fields.
 - You can optionally choose to upload a credential wallet .zip file. This file contains client credentials and is generated by the database.
 - To use blockchain tables to store the rich history database, select **Use Database Blockchain Table**. The underlying database must support blockchain tables.
 - To specify table and row retention by using a data definition language (DDL) statement, select Advanced Configuration Query and then enter the DDL statement.

Select **Save**.

6

Create an Instance

Deploying the stack creates Blockchain Platform Manager, which you can use to provision, configure, and manage instances of Oracle Blockchain Platform Enterprise Edition for Besu. Instance nodes run on a set of available Kubernetes worker nodes.

You use Blockchain Platform Manager to complete the following tasks:

- Create, start, stop, and delete instances of Oracle Blockchain Platform Enterprise Edition for Besu.
- View details about all instances and activity records for all instances.
- Configure, activate, and use the LDAP service or configure your own OIDC provider for the installation.
- Create and save user accounts (OpenLDAP users only).

To create an instance:

1. Open Blockchain Platform Manager (<https://controlplane.domainname/console/index.html>)
2. Log in to Blockchain Platform Manager by using your Instance Admin role user you have created in your identity provider.
3. On the **Instances** page, click **Create Instance** to create an instance of the Besu network. The Besu Create Instance dialog is displayed.
4. Complete the following fields:

Field	Description
Instance Name	Enter a name for your Oracle Blockchain Platform instance. The service instance name: • Must contain one or more characters. • Must not exceed 15 characters. • Must start with an ASCII letter: a to z. • Must contain only lower case ASCII letters or numbers. • Must not contain a hyphen. • Must not contain any other special characters. • Must be unique within the identity domain.
Description	Optional. Enter a short description of the Oracle Blockchain Platform instance.
Domain Name	Enter the domain name for the cluster that you specified when installing. This must contain only lower case ASCII letters. The hostnames generated for the Blockchain Instance services make use of the domain name and the instance name as parent domain and sub domain respectively.

Field	Description
Role	Select Founder to create a complete blockchain environment. This instance becomes the founder organization and you can onboard new participants in the network later. Select Participant to create an instance that will join an existing blockchain network created elsewhere before this instance can be used.
Instance Access Group	The group name created in the identity provider containing users who are allowed to access and operate with this instance.
Configuration	Select a provisioning shape which meets the needs of your deployment. Only Enterprise is currently supported for Oracle Blockchain Platform Enterprise Edition for Besu.
Chain ID	(Founder instance only) Enter your Ethereum chain ID. The default is 1500. Provide a unique numeric identifier for this network avoiding common public IDs like 1, 137, 1337, or 56, to ensure transaction signatures are valid only on this specific Besu deployment and protected from replay attacks.
Third Party Root CA	If you want to use certificates from your own certificate authority and use the Oracle Blockchain Platform certificate authority as an intermediary CA, you can upload your CA archive. The certificate that you upload will be used to sign the intermediary certificates for Oracle Blockchain Platform nodes, thus including them under your root CA chain. The archive is a .zip file which contains the following files: - CA chain: This file must be named ca-chain.pem. The entire CA file sequence from the signing CA to the top-level CA must be present. - key: This file must be named ca-key.pem. A 256-bit key that uses the prime256v1 curve is recommended. The key file must be an unencrypted private key in the PKCS #8 format. - certificate: This file must be named ca-cert.pem. The certificate file must be in Base64 format and must include the subject key identifier extension. The archive must be less than 2 MB. The files must directly reside inside the .zip archive such that when the archive (.zip) is extracted, the files are visible in the current directory at the same level as the archive (.zip) file. The files must not be inside a nested directory inside the archive. The product was tested with an X.509 v3 certificate with the following properties. Key type: Elliptic curve cryptography (ECC) Curve name: prime256v1 Certificate signing algorithm: ecdsa-with-SHA256
Block Period	(Founder instance only) Minimum time (in seconds) between new block creation.
Validator Node Count	(Founder instance only) Number of nodes responsible for validating and proposing blocks.
BootNode Count	(Founder instance only) Number of nodes used for initial peer discovery.
RPC Node Count	(Founder instance only) Number of nodes providing APIs for external communication.

Field	Description
Archive Node Count	(Founder instance only) Number of archive nodes to provision.
Smart Contract Governance Mode	Select Governed to manage smart contracts through a Digital Assets administrative user. Select NoOp to deploy smart contracts without governance controls.
Rich History Oracle DB Profile	Select your rich history database profile. It must be created and configured before creating the instance.
Upload Genesis File	(Participant instance only) Upload the genesis metadata .zip file from the founder instance. Select Export Genesis Metadata from the founder instance Actions menu to download the genesis metadata file from the founder.

The instance creation process runs, creating the pods for the nodes and services that make up the Besu network.

- Once the instance is listed in the Up state, add the Hosts entries to your Hosts file.

```
<ingress-ip> console.<instance-name>.<instance-domain> rpcproxy.<instance-name>.<instance-domain> wrapper.<instance-name>.<instance-domain>
auth.<instance-name>.<instance-domain>
```

You can retrieve the domain name from the Application Information pane in OCI: [Application Information](#).

Verify Your Instance

Navigate to your service in Blockchain Platform Manager and sign in to verify that your instance is up and running.

- Use the service console path to access the blockchain platform service console.

```
https://console.<instance-name>.<domain-name>
```

- Log in to the service console with the instance administrator credentials.

Note

Once you've successfully logged into Blockchain Platform Manager with this user and provisioned an instance, you may want to disable the default admin user (obpadmin) for security reasons. This can be done from the Configuration page Platform Manager Settings tab.

Now that your instance is created, you can explore the pages in the service console:

- Dashboard: Overview of your instance health and activity.
- Nodes: List and manage Besu network nodes.
- Explorer: View and search transactions, blocks and accounts with rich history-backed activity.
- Contracts: View deployed smart contracts, inspect versions and activity, and manage contract operations.

- Digital Assets: View and download pre-built business application packages, tokenized frameworks, and view digital assets deployed on the instance.
- Developer Tools: View and download token standard and protocol libraries, pre-deployed contracts, and Oracle's Blockchain App Builder for Hyperledger Besu.
- Accounts: View user's accounts, including transaction history and registered asset balances.
- Logs: View instance component logs by pod or container and download as needed for troubleshooting.

For more information on the console and using it to manage your network, see [Manage the Organization and Network](#).

Service Endpoints

These are the service endpoints generated once your instance is created.

The following table shows the endpoints for Blockchain Platform Manager, the service console, and the RPC proxy.

Service	Endpoint Type	URL Format	URL Source
Blockchain Platform Manager	Web URL	<code>https://controlplane.<domain-name>/console/index.html</code> Example: <code>https://controlplane.obpee.example.com/console/index.html</code>	The domain name that you specified when installing Blockchain Platform Manager.
Service console	Web URL	<code>https://console.<instance-name>.<instance-domain-name></code> Example: <code>https://console.obpee.example.com</code>	The instance name and domain name that you specified when creating the instance.
RPC proxy	API endpoint	<code>https://rpcproxy.<instance-name>.<instance-domain-name>:443</code> Example: <code>https://rpcproxy.obpee.example.com:443</code> <code>https://rpcproxy.obpee.example.com:443/v1/besu/query</code>	The instance name and domain name that you specified when creating the instance. A detailed list of APIs is available at the RPC proxy URL in the <code>/openapi</code> path.
Authentication service	Fetch access token	<code>https://auth.<domain-name>:443//auth/realms/master/protocol/openid-connect/token</code> Example: <code>https://auth.obpee.example.com:443//auth/realms/master/protocol/openid-connect/token</code>	When using OpenLDAP, the domain name that you specified when installing Blockchain Platform Manager. For OIDC, the endpoint is specific to the OIDC provider. IDCS has its own token fetch endpoint.

These endpoints can also be found in Blockchain Platform Manager. On the **Instances** page, select the action menu and select **Instance URLs**.

Create an Organization and Wallet

Once your Blockchain Platform instance is provisioned, you can create and configure your organization, add users to it, and create a wallet.

Create Your Organization and Wallets

1. Open Blockchain Platform Manager (<https://controlplane.domainname/console/index.html>).
2. Log in to Blockchain Platform Manager by using the username and password that you added when creating your identity provider. Note that this user must be a member of the Wallet Super Admin group in order to create an organization.
3. On the **Wallet** page **Organization** tab, select **Create Organization**.
4. Enter the organization information, and select **Create Organization**.
5. Associate your instance with the new organization. Expand the newly created organization, and select **Manage Organization Access**. Select which instances are associated with the new organization by selecting them from the **Access Scope** list, and select **Add Organization Access Mapping**.
6. Select the **Users** tab, and select **Register User**.
7. Enter all the information for the user. Select which organization they belong to, and select **Register User**. Repeat for each user.
 - Note that these users must exist in the identity provider.
 - For OpenLDAP, User ID must match the name claim in the JSON Web Token.
 - For IDCS, User ID must match the sub claim in the JSON Web Token.
8. If you're using policies to govern how digital asset keys, transactions, and user compliance requirements are managed, on the **Policies** tab, select **Add Policy** and enter the policy details.
9. On the **Wallets** tab, select **Create Wallet**. Enter the wallet details for a user, and select **Create Wallet**. Repeat for each user.
10. For instances that use governance (Governed mode), you must call the governance contract to register the wallet. For more information, see [Create Wallet](#) in the UWaaS API documentation. The following `curl` command shows an example of this wallet registration API call.

```
curl --location --request POST 'https://uwaas.example.com/v1/uwaas/wallets' \
--header 'Authorization: Bearer eZjcd7QIrxI8i01wpQ' \
--header 'Content-Type: application/json' \
--data-raw '{
  "orgId": "TestOrg",
  "orgNetworkId": 4,
  "userId": "e194554b-bc47-5ae7-b052-76db72f8dd09",
  "walletLabel": "ha1",
  "walletType": "1",
  "status": "ACTIVE",
}
```

```
"walletTags": ["ha1"]  
}'
```

Universal Wallet as a Service (UWaaS)

Universal Wallet as a Service (UWaaS) is a custodial wallet platform that provides standardized, scalable APIs for managing digital wallets, enforcing business policies, and executing blockchain transactions according to defined operational rules.

The UWaaS wallet service consists of three components:

- Policy and Rule Manager.
- UWaaS Wallet Manager.
- Instance or Local Wallet Service.

Policy and Rule Manager

The Policy and Rule Manager provides APIs for managing the lifecycle of business policies and defining their operational scope. These APIs support creating, enabling, disabling, editing, and deleting business policies and rules.

UWaaS Wallet Manager

The UWaaS Wallet Manager models the functional and operational processes of a business through the following logical objects:

- Organization.
- Network partitions.
- Organization network partitions.
- Wallets.

Organization

The Organization logical object maps to internal departments of a business or to members of a business consortium. This aligns business functions with logical organization objects, such as finance and purchasing departments or consortium members.

Network Partitions

The Network Partition object defines local partitions or networks that encapsulate existing physical blockchain networks, whether public, private, or permissioned.

Organization Network Partitions

The Organization Network Partition object associates network partitions with organizations and enforces an operational boundary for each function. These objects can be associated with policies and rules created by the Policy and Rule Manager.

Wallets

Business administrators create wallet objects under an organization on behalf of end users. Each wallet is uniquely identified by a UWaaS UUID and encapsulates the physical wallet address for a specific blockchain network. The wallet object provides an API for retrieving the underlying physical wallet address.

Instance or Local Wallet Service

The Instance or Local Wallet Service provides APIs for querying and transacting on the instance network by using the signing, transaction, query, and broadcast capabilities of the

service. Because it is embedded within the instance network, these APIs accept a physical wallet address instead of a wallet UUID in this Limited Availability release.

8

Manage the Organization and Network

The Oracle Blockchain Platform Service Console is a comprehensive web interface that enables users to monitor blockchain networks and perform routine administrative tasks. The console also provides specialized tools and functionality for smart contract developers and wallet administrators.

Oracle Blockchain Platform Service Console Overview

After an Oracle Blockchain Platform instance is provisioned, all core functions required to interact with and manage the blockchain network are accessible through the service console. Key capabilities include:

- Managing blockchain nodes
- Configuring network settings and policies
- Deploying smart contracts
- Creating and accessing account wallets
- Using the ledger browser to inspect block details and transactions
- Accessing log files (administrators and operators only)

The console's graphical user interface organizes these features across several tabs:

- **Dashboard:** Presents a summary overview of network health, node status, latest block height, and transaction volume.
- **Nodes:** Displays detailed information about all active nodes, with menus for node management actions.
- **Explorer:** Displays a block explorer to list recent blocks, search by block number and transaction hash, and view details of the transactions.
- **Contracts:** Displays the deployed contracts, and lets you manage them.
- **Digital Assets:** All of the digital assets functionality is available from the Digital Assets page in the console.
- **Developer Tools:** Displays predeployed smart contracts, including the source code, Application Binary Interface (ABI), and deployment address.
- **Accounts:** Enables creation and management of Externally Owned Account (EOA) wallets.
- **Logs:** Provides access to log files generated by each pod running Besu nodes.

Note: In the Limited Availability release, some tabs might not yet be connected to back end services. For operations such as deploying smart contracts, users can utilize the provided RPC Proxy APIs, which act as a Web3 API gateway.

Dashboard

The dashboard shows the following details about your instance.

Summary Header

- Number of nodes

- Total Transactions
- Total number of blocks
- Total number of nodes

Besu Network Information

- Version
- Consensus algorithm
- Chain ID
- Block time

Kubernetes Cluster Details

- Worker node details
- CPU resources
- Memory resources
- Storage resources using OKE persistent volume claims (PVC) disks on Block Storage

Nodes

The Nodes views show a summary by node type and details, which include Name, Organization, Type, Node ID, and Status. There is an **Actions** menu available for each node action to display the details and, if needed, restart the node.

Explorer

The Explorer tab provides a block explorer to list recent blocks, search by block number and transaction hash, and view details of the transactions.

Contracts

You can view deployed smart contracts, inspect them, and manage contract operations from this page.

Digital Assets

After you create an instance, click the **Digital Assets** tab to open the Digital Assets page. You can access the following panes from the Digital Assets page.

Wholesale CBDC Application

You can use the Wholesale CBDC Application pane to complete the following tasks. For more information, see Wholesale Central Bank Digital Currency Application.

- Preview sample actions for the application without actually calling a smart contract.
- Download the prebuilt wholesale CBDC smart contract.
- Download the wholesale CBDC wrapper APIs.
- Download database view definitions, which you can use for analytics and the prebuilt web application.
- Download the sample web application.

Deposit Token Application

You can use the Deposit Token Application pane to complete the following tasks.

- Preview sample actions for the application without actually calling a smart contract.
- Download the prebuilt tokenized deposit smart contract.
- Download the tokenized deposit wrapper APIs.
- Download database view definitions, which you can use for analytics and the prebuilt web application.
- Download the sample web application.

Accounts

The **Accounts** view provides access to the custodial, multi-partition Wallet Service within the Oracle Blockchain Platform Console. This service enables you to perform essential wallet operations, including:

- Creating up to three wallets per user
- Viewing all wallets owned by the user
- Suspending and reactivating wallets
- Signing transactions with a selected wallet

These wallets are custodial, and designed for use with the Besu blockchain network. You can create new wallets directly from the console, or upload existing wallet keys (by using `.pem` files) to associate external wallets with your account.

Currently, the **Accounts** view supports only a single partition that is linked to the Besu instance.

Logs

The **Logs** view allows you to tail or view the last N lines of logs from each container in any pod in the namespaces of the underlying Kubernetes cluster.

You can search logs by instance, pod, and container. You can also specify the number of lines to display and apply filter expressions to help identify specific log messages.

You can also download log files directly from the console. For advanced log management, you can configure log aggregation using persistent logging with Oracle Cloud Infrastructure (OCI) or integrate with external tools such as `fluentd` for a unified logging approach.

9

RPC Proxy

The RPC proxy is the primary interface for client applications. It forwards requests to the configured upstream node endpoints.

The RPC proxy provides an authenticated and authorized access layer for Ethereum JSON-RPC requests. It also provides specialized features for signing transactions, running queries, and managing wallets.

Authentication

Secured endpoints require a bearer token in the authorization header:

Authorization: Bearer <access_token>

The proxy validates the token in accordance with the deployment configuration (for example, signature validation, expiration, issuer, and audience).

You can use the following command to generate a bearer token by using the access token endpoint that is available in Blockchain Platform Manager.

```
curl -sS -X POST \
  "<Access Token Endpoint - fetched from Blockchain Platform Manager>" \
  -H "Content-Type: application/x-www-form-urlencoded" \
  -d "grant_type=password" \
  -d "client_id=<CLIENT_ID>" \
  -d "client_secret=<CLIENT_SECRET>" \
  -d "username=<USERNAME>" \
  -d "password=<PASSWORD>"
```

The following text shows an example response. You then use the value of `access_token` as the bearer token for subsequent calls.

```
{
  "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9....",
  "expires_in": 300,
  "token_type": "Bearer"
  ...
}
```

Authorization

Access is controlled using the following group membership, which is conveyed in the token claims. See [User Groups and Roles](#).

OpenAPI Specification

The RPC proxy publishes an OpenAPI specification for its REST endpoints. The following text shows an example request.

```
curl -sS \  
  "<RPC_PROXY_BASE_URL>/openapi" \  
  -H "Authorization: Bearer <access token>" \  
  -H "Accept: application/yaml"
```

The following types of APIs are exposed by the RPC proxy. For full details see the OpenAPI specification at [Web3 Transactions and Besu Instance APIs](#).

- JSON-RPC proxy API: Provides an authenticated and authorized entry point for Ethereum JSON-RPC methods such as `eth_call`, `eth_getBalance`, `eth_sendRawTransaction`, and `eth_getTransactionReceipt`.

Develop and Use Smart Contracts

Use Oracle Blockchain Platform Enterprise Edition for Besu to deploy the bundled applications, download sample Solidity contracts, work with tokenized assets, and call Solidity methods for account and token contracts.

- [One-Click Application Deployment](#)
 - Use the one-click function in Blockchain Platform Manager to deploy an application to your instance.
- Smart Contracts for Digital Assets
 - Use the provided digital assets SDK packages to work with tokenized assets: one extends the ERC-20 standard and one extends the ERC-1155 standard. For more information, see [Install Smart Contracts for Digital Assets](#) in the Digital Assets Guide.
- Solidity Smart Contract API for Digital Assets
 - Call Solidity methods that support tokens based on the ERC-20 and ERC-1155 standards. For more information, see [Digital Assets APIs](#) in the Digital Assets Guide.
- [Sample Smart Contracts](#)
 - Find sample contracts on the Developer Tools page in the service console. Download the Solidity source code or the application binary interface (ABI).
 - Use the ABI with RPC proxy API calls to interact with deployed smart contracts.
 - Import the source into a Solidity development environment (for example, Hardhat or Remix) to modify or extend the contracts.

One-Click Application Deployment

After you satisfy the prerequisites, you can deploy one of the bundled digital assets applications from Blockchain Platform Manager.

Prerequisites

The following steps outline the prerequisite workflow for one-click deployment.

1. Install Oracle Blockchain Platform Enterprise Edition. For more information, see [Provisioning Oracle Blockchain Platform Enterprise Edition](#) from Oracle Cloud Marketplace.
2. Configure your identity provider, either the integrated OpenLDAP server or Oracle Identity Cloud Service (IDCS).
 - User Management with OpenLDAP
 - User Management with an External Identity Provider
3. Create a rich history database configuration. For more information, see [Create a Rich History Database Configuration](#).

Note

While the rich history function supports both Oracle AI Database and MySQL, you must use Oracle AI Database with the bundled digital assets applications.

4. Create an instance that uses the rich history configuration. For more information, see [Create an Instance](#).
5. On the **Wallet** page of Blockchain Platform Manager, configure an organization, users, and wallets. For more information, see [Configure a Wallet](#).
 - a. Create an organization.
 - b. Create a user (for the logged-in user or the user who will deploy the application) and add the user to the organization.
 - c. Create a wallet for the logged-in user or the user who will deploy the application.
6. For an instance in Governed mode, make a transaction call to the governance contract to register the wallet to allow the deployment to succeed. Use an API signature with the following form, replacing the variables, server, and bearer token with the values from your instance.

```
curl --location --request POST 'https://uwaas.example.com/v1/uwaas/wallets' \
--header 'Authorization: Bearer fZjcd7QIryI8i01wpQ' \
--header 'Content-Type: application/json' \
--data-raw '{
  "orgId": "TestOrg",
  "orgNetworkId": 4,
  "userId": "e194554a-bc46-4ae7-a152-75db72f8dd09",
  "walletLabel": "haha1",
  "walletType": "1",
  "status": "ACTIVE",
  "walletTags": ["haha1"]
}'
```

One-Click Deployment

1. On the **Instances** page of Blockchain Platform Manager, select the menu for the instance that you created.
2. Select **Deploy Application**.
3. Select the application to deploy from the **Application** list.
4. Enter a deployment name in the **Deployment Name** field. Deployment names can use only lowercase letters, numbers, and hyphens, and must start and end with only either a letter or number. Select **Next**.
5. In the Smart Contract Deployment window, select **Deploy new contract** or **Use existing contracts**. Select **Use existing contracts** to enable a participant instance to join the same blockchain as a founder instance where the application is already deployed.
 - For a new deployment, enter a **Token Name**, **Token Symbol**, and **Organization**.
 - For existing contracts, enter the **Token Contract Address**, **Account Contract Address**, and **Organization**.

Click **Next**. The contract is deployed, the Oracle REST Data Services (ORDS) artifacts are deployed, and the wrapper API is registered.

6. In the Review Deployment window, ensure that the application and deployment name are correct. Select **Submit**.
7. A deployment success message is displayed, as well as a link to access the application.
8. Update your `/etc/hosts` file to add an entry for `<sample-app-deployment-name>.<instance-name>.<instance-domain>` mapped to the Istio ingress gateway port.
9. If you use IDCS via OpenID Connect, add the redirection URL for the bundled digital assets application to the confidential client application that you created when you configured IDCS.

You can now use the provided link to configure and use the smart contract application.

For more information about the applications, see Wholesale Central Bank Digital Currency Application and Deposit Token Application in the Digital Assets Guide.

Sample Smart Contracts

Oracle Blockchain Platform Enterprise Edition for Besu includes predeployed sample smart contracts.

To access the predeployed contracts, click **Developer Tools** in the service console, and then click **Pre-deployed Contracts**.

Select the **Download** menu for each contract to download the Solidity source code or the application binary interface (ABI). You can use this information to interact with deployed smart contracts by using RPC proxy API calls. You can also import the source code into a Solidity development environment such as Hardhat or Remix and make changes or enhancements as needed.

Hash Time-Locked Contract (HTLC)

This contract is deployed when you create an instance. The Hash Time-Locked Contract enables atomic exchanges of ETH and ERC-20 tokens, where a receiver claims a payment prior to a deadline or the payment is refunded to the sender. The contract supports the following operations.

- Exchange ETH or ERC-20 tokens.
- Lock funds for a receiver under a secret condition for a specified amount of time.
- Claim locked funds by providing the correct secret before the lock expires.
- Refund unclaimed locked funds to the sender after the lock expires.
- Get the status of a locked exchange, including whether it has been claimed or refunded, the locked amount, and the lock deadline.

LockBox

This contract is deployed when you create an instance. The LockBox contract implements a time-locked vault that holds ETH and ERC-20 tokens and then releases the contents to a designated owner at a specific time. The contract supports the following operations.

- Deposit funds into the vault by sending ETH directly or transferring ERC-20 tokens.
- Get vault details, including who the beneficiary/owner is, what the release time is, and what tokens/ETH are in the vault. Any user can get this information.
- Release funds at a specified time. After the specified time stamp, the owner can call the release action to retrieve the funds.

A

Application Information

After creating a stack in Oracle Cloud Infrastructure, all information about the created stack is available in its Application Information tab. You can get to this page from the OCI Resource Manager.

Table A-1 General

Variable Name	Description	Default
Title	Clickable text that redirects the user to the marketplace listing.	Oracle Blockchain Platform on Oracle Cloud Infrastructure Container Engine for Kubernetes
Description	Simple text describing the stack.	The stack deploys a new OCI Kubernetes Engine Cluster and hosts the Blockchain Platform on it
Informational Text	Simple informational text to guide the user once they have run Apply on the stack.	After deployment, copy the Domain Host Entry and add it to your client machine's hosts file. For created instances, append the hosts entry with <pre>console.<instance-name>.<instance-domain> rpcproxy.<instance-name>.<instance-domain></pre> Navigate to the Blockchain Platform Manager for creating instances.

Table A-2 Blockchain Platform Details

Variable Name	Description	Default
Ingress Gateway IP	IP address of the gateway created by the stack's OKE Cluster, used for DNS resolution.	
Domain Host Entry	Entry to append to your hosts file. • MacOS and Linux: /etc/hosts • Microsoft Windows: C:\Windows\System32\drivers\etc\hosts	<pre><ingress-ip> console.<instance-name>.<instance-domain> rpcproxy.<instance-name>.<instance-domain> wrapperapi.<instance-name>.<instance-domain> auth.<instance-name>.<instance-domain></pre> Example: 203.0.113.204 console.example.com rpcproxy.example.com wrapperapi.example.com auth.example.com

Table A-2 (Cont.) Blockchain Platform Details

Variable Name	Description	Default
Blockchain Platform Manager	URL to Blockchain Platform Manager on Kubernetes.	https:// controlplane.<instance- domain>/console/index.html Example: https:// controlplane.example.com/ console/index.html

Table A-3 Kubernetes Cluster Details

Variable Name	Description	Default
OKE Cluster Name	OKE cluster name, with the dynamic deployment ID of the stack apply job appended	<cluster-name> (<deployment-id> Example: TestCluster01 (tusf), TestCluster02 (Gd45)
OKE Cluster Compartment OCID	Compartment OCID of the OKE cluster	Example: ocid1.compartment.oc1..aaa aaaaa4wsjx5rugizog2i2bqlvd 7ppcz7xm4nybdsb5vgrwqsxig t2sia
OKE Cluster OCID	OCID of the OKE cluster	Example: ocid1.cluster.oc1.eu- milan-1.aaaaaaa5tkfntba7n cdx774gl7yq4qvfhgwtncfy4cd uglmlck7jkm3y2q
Kubernetes Configuration	The Kubernetes configuration of the cluster. Note: OCI CLI is necessary for the authentication for this configuration on the machine where it is used.	

Table A-4 Terraform Deployment Details

Variable Name	Variable Name	Description
Deployment Id	Randomly generated 4 character deployment ID, unique for each terraform job triggered.	Example: tusf, Gd45
Deployed Region		Corresponding region where the stack and its resources have been deployed (created).
Stack Version		Version of the stack.

B

Connect to Oracle Kubernetes Engine

You can use the OCI command line interface (CLI) to access an Oracle Kubernetes Engine (OKE) cluster.

The following instructions assume that a kubeconfig file is already available.

1. Install the OCI CLI.

- On Linux or macOS, enter the following command:

```
bash -c "$(curl -L https://raw.githubusercontent.com/oracle/oci-cli/master/scripts/install/install.sh)"
```

- On Microsoft Windows, use the .msi installer linked to from [Installing the CLI](#).

For more information about the OCI CLI, see [Working with the CLI](#).

2. Install kubectl.

- On macOS, enter the following homebrew command.

```
brew install kubectl
```

- On Linux, enter the following curl command:

```
VERSION=$(curl -L -s https://dl.k8s.io/release/stable.txt)curl -LO "https://dl.k8s.io/release/${VERSION}/bin/linux/amd64/kubectl"; chmod +x kubectl && sudo mv kubectl /usr/local/bin/
```

- On Microsoft Windows, use the following Chocolatey command.

```
choco install kubernetes-cli
```

For more information about installing kubectl, see [Install Tools](#) in the Kubernetes documentation.

3. Enter the following command to configure your OCI CLI credentials.

```
oci setup config
```

Provide the following information to the command.

- The tenancy OCID
- Your user OCID
- The region (for example, ap-mumbai-1)
- Your API key (fingerprint/private key path)

To get a new API key, enter the following command, and then select **Identity & Security**, and then **Users**, and then **Your User Account**, and then **API Keys** to upload the public key to the console.

```
oci setup keys
```

4. Add the kubeconfig file to the default path, so that it is accessible by kubectl.

- On macOS and Linux, add the kubeconfig file to the following directory.

```
~/.kube/config
```

- On Microsoft Windows, add the kubeconfig file to the following directory.

```
%USERPROFILE%.kube\config
```

5. Generate the kubeconfig file for the Kubernetes cluster.

- a. From the cluster page in OCI, select **Actions**, and then **Access cluster**. The Access cluster window is displayed.
- b. Select **Local access**.
- c. Select **Copy** for **Public endpoint access command**.
- d. Run the copied command to generate the kubeconfig file.

6. Run the following commands to validate connectivity to the cluster.

```
kubectl cluster-info
kubectl get nodes -o wide
kubectl get ns
```

Private endpoint clusters require network access to the virtual cloud network (VCN). Public endpoint clusters require internet connectivity to the API server endpoint.

7. If you use multiple OCI profiles, set the following environment variable.

- Linux and macOS: `export OCI_CLI_PROFILE=`
- Microsoft Windows: `$env:OCI_CLI_PROFILE=""`

If kubectl returns a 401 or 403 error, verify that your identity and access management (IAM) policies allow you to use and manage a cluster family in the compartment, and that your kubeconfig file was generated for the correct cluster and region.

To scale your OKE cluster, complete the following steps.

1. In the OCI console, select your OKE cluster and then select **Node pools**.
2. Select the node pool that you want to scale.
3. From the **Actions** menu, select **Edit**. The Edit node pool page is displayed.
4. Make changes as needed to the node count or node shape. To change the node shape (which affects the number of OCPUs and the amount of memory), select **Change shape**.
5. Select **Update** to confirm your changes.
6. From the **Actions** menu, select **Cycle nodes** to apply the updated configuration. The Cycle nodes page is displayed.

7. Under **Cycling options**, select **Replace nodes**. Enter numbers for **Maximum surge** and **Maximum unavailable**, then select **Cycle nodes**. When the state of the node pool is updated to **Active**, the scaling is complete.

Alternately, if you installed by deploying a stack, you can click **Destroy** to delete your current stack in the OCI console. and then select **Edit**, and then **Edit stack** to edit the stack definition to change the node count or node shape as needed. Select **Run apply** and **Save changes** to create a stack that uses the new parameters.

Glossary

Index