

SOLUTION PARTNER FOR **SMART TECHNOLOGY**



Installation

LENA Support

Version 1.3.3.0

Table of Contents

1. OVERVIEW.....	1
1.1. Components.....	1
1.1.1. Server.....	1
1.1.2. Agent, Advertiser.....	1
1.1.3. Manager.....	1
1.2. Mechanism.....	2
1.3. Provided Assets.....	5
1.4. System Requirements.....	5
2. Architecture Decisions.....	6
2.1. Process.....	6
2.2. Container Operating Environment Considerations.....	7
2.2.1. Container Platform Operating Environment Characteristics.....	7
Kubernetes.....	8
ECS.....	12
Container Platform Characteristics Summary.....	13
2.2.2. LENA Server Type Operating Methods.....	14
2.3. Common Considerations.....	14
2.3.1. OS.....	14
2.3.2. JDK.....	14
2.3.3. Execution User.....	15
2.3.4. Library.....	15
2.4. Server Type Considerations – Manager.....	16
2.4.1. Deployment.....	16
2.4.2. Specifications.....	16
Memory.....	17
Disk.....	17
2.4.3. Configuration.....	17
Network.....	17
Environment Variables.....	18
Directory Structure.....	18
Log & Dump Output.....	19
Health Check.....	19
2.5. Server Type Considerations – Session Server.....	20
2.5.1. Deployment.....	20
2.5.2. Specifications.....	21
Memory.....	21
Disk.....	21
Configuration.....	21
Network.....	21
Environment Variables.....	22

Directory Structure	23
Log	23
Health Check	23
2.6. Server Type Considerations – WAS	24
2.6.1. Deployment	24
2.6.2. Specifications	24
Memory	24
Disk	25
2.6.3. Configuration	25
Network	25
Environment Variables	25
Directory Structure	27
Log & Dump Output	28
Health Check	29
Server Configuration Management	30
Container Image Build	31
Application Deployment	31
2.7. Server Type Considerations – Embedded WAS	32
2.7.1. Deployment	32
2.7.2. Specifications	32
Memory	32
Disk	33
2.7.3. Configuration	33
Network	33
Environment Variables	33
Directory Structure	35
Log & Dump Output	35
Health Check	36
Container Image Build	36
Application Deployment	37
2.8. Server Type Considerations – Web Server	37
2.8.1. Deployment	37
2.8.2. Specifications	37
Memory	38
Disk	38
2.8.3. Configuration	38
Network	38
Environment Variables	38
Directory Structure	40
Log	41
Health Check	41
Server Configuration Management	41

Container Image Build	41
3. Installation Common Requirements	42
3.1. Installation Preparation	42
3.2. Base Image Creation	42
3.2.1. Base Image Creation Procedure	43
Dockerfile Creation	43
Dockerfile Creation (Regular Account)	46
Docker Image Build	47
Docker Image Registration	47
4. Kubernetes-based Deployment	49
4.1. Common Deployment Considerations	49
4.1.1. Deployment Preparation	49
4.1.2. Deployment Execution	49
Setting the Working Namespace	49
Kubernetes Resource Deployment and Updates	49
Deleting Kubernetes Resources	50
Workload Updates and Rollbacks	50
Verifying Deployed Resources	50
4.2. Manager Deployment	51
4.2.1. Configuration Items	51
Basic Configuration Items	51
Application-Time Decision Items – Workload Related	51
Application-Time Decision Items – Service Related	52
4.2.2. Manifest-based Deployment	52
Workload	52
Service	54
4.2.3. Manager Access	55
4.3. Session Server Deployment	56
4.3.1. Deployment Preparation	56
4.3.2. Configuration Items	56
Basic Configuration Items	56
Application-Time Decision Items – Workload Related	56
Application-Time Decision Items – Service Related	57
4.3.3. Manifest-based Deployment	57
Workload	57
Service	59
4.3.4. Server Registration Verification	60
4.4. WAS Deployment	60
4.4.1. Deployment Preparation	60
4.4.2. Configuration Items	60
Basic Configuration Items	60
Application-Time Decision Items – Workload Related	60

Application-Time Decision Items – Service Related	62
4.4.3. Manifest-based Deployment	62
Workload.....	62
Service.....	64
4.4.4. Server Registration Verification.....	64
4.5. Embedded WAS Deployment	65
4.5.1. Deployment Preparation.....	65
4.5.2. Configuration Items.....	65
Basic Configuration Items	65
Application-Time Decision Items – Workload Related	65
Application-Time Decision Items – Service Related	66
4.5.3. Manifest-based Deployment	66
Workload.....	66
Service.....	68
4.5.4. Server Registration Verification.....	69
4.6. Web Server Deployment	69
4.6.1. Deployment Preparation.....	69
4.6.2. Configuration Items.....	69
Basic Configuration Items	69
Application-Time Decision Items – Workload Related	69
Application-Time Decision Items – Service Related	71
4.6.3. Manifest-based Deployment	71
Workload.....	71
Service.....	73
4.6.4. Server Registration Verification.....	73
5. ECS-based Installation	75
5.1. ECS Overview.....	75
5.2. Installation Preparation	75
5.3. Manager Deployment.....	75
5.3.1. Configuration Items.....	75
Basic Configuration Items	75
Decision Items for Application Timing	76
5.3.2. Task Configuration	76
Task Definition	77
Volume Addition	77
Container Addition	77
Environment Variable Configuration.....	77
Health Check Configuration.....	78
Volume Mapping	78
5.3.3. Service Configuration	78
Service Definition.....	78
Service Discovery Configuration	79

5.3.4. Service Startup and Verification	79
Service Status Verification	79
Task Status Verification	79
5.4. Session Server Deployment	80
5.4.1. Deployment Preparation	80
5.4.2. Configuration Items	80
Basic Configuration Items	80
Decision Items for Application Timing	80
5.4.3. Task Configuration	81
Task Definition	81
Container Addition	82
Environment Variable Configuration	82
5.4.4. Service Configuration	82
Service Definition	82
Service Discovery Configuration	83
5.4.5. Service Startup and Verification	83
Service Status Verification	83
Task Status Verification	84
5.5. WAS Deployment	84
5.5.1. Deployment Preparation	84
5.5.2. Configuration Items	84
Basic Configuration Items	84
Decision Items for Application Timing	84
5.5.3. Task Configuration	85
Task Definition	86
Container Addition	86
Environment Variable Configuration	86
5.5.4. Service Configuration	87
Service Definition	87
Service Discovery Configuration	87
5.5.5. Service Startup and Verification	87
Service Status Verification	87
Task Status Verification	88
5.6. Embedded WAS Deployment	88
5.6.1. Deployment Preparation	88
5.6.2. Configuration Items	88
Basic Configuration Items	88
Decision Items for Application Timing	88
5.6.3. Task Configuration	89
Task Definition	89
Container Addition	90
Environment Variable Configuration	90

5.6.4. Service Configuration	90
Service Definition.....	90
Service Discovery Configuration	91
5.6.5. Service Startup and Verification	91
Service Status Verification	91
Task Status Verification	92
5.7. Web Server Deployment	92
5.7.1. Deployment Preparation.....	92
5.7.2. Configuration Items	92
Basic Configuration Items	92
Decision Items for Application Timing	92
5.7.3. Task Configuration	94
Task Definition	94
Container Addition	94
Environment Variable Configuration.....	94
5.7.4. Service Configuration	95
Service Definition.....	95
Service Discovery Configuration	95
5.7.5. Service Startup and Verification	96
Service Status Verification	96
Task Status Verification	96
6. VM/Host-based Installation	97
6.1. Installation Preparation	97
6.2. LENA Installation.....	97
6.3. Directory Structure.....	97
6.4. Manager Installation.....	99
6.4.1. Manager Installation	99
6.4.2. Manager Execution	100
6.5. Node Agent Execution.....	102
6.5.1. Node Agent Execution	102
6.5.2. Node Agent Operation Check	103
6.5.3. Node Agent Stop.....	103
6.6. Session Server Installation (WEB UI-based)	104
6.6.1. Session Server Installation.....	105
6.6.2. Server Start	105
6.6.3. Server Delete	105
6.6.4. Server Registration	106
6.7. Session Server Installation (CLI-based).....	106
6.7.1. Session Server Installation.....	106
6.7.2. Session Server Start	109
6.7.3. Session Server Delete	110
7. Appendix.....	112

7.1. LENA Supported Spec Versions	112
7.2. Manager DB File Backup	112
7.3. Deletion of Manager Internal History	112
7.4. Manager admin Password Reset	112
7.5. Recommended OS Parameters for LENA Installation (CentOS).....	113
7.6. LENA Files that Grow Periodically	114
7.7. WAS Image OS Reference.	115

Chapter 1. OVERVIEW

This document describes the architectural decision factors and installation procedures required before operating Container-based LENA Servers. For comprehensive information about LENA's full functionality and operations, please refer to the separately provided operator manual.

This document is based on LENA version 1.3.1c and includes the following content:

- LENA for Container architectural decision factors
 - Considerations for deployment environments
 - Considerations by server type
- LENA for Container installation
 - Base Image Build
 - Kubernetes-based installation
 - Docker-based installation
 - ECS-based installation
 - VM / Host-based installation (Manager, Session Server)

1.1. Components

LENA consists of Web Server, Application Server, Session Server, Node Agent that monitors Web Server status, Advertiser installed on Application Server to provide status information, and Manager which is an integrated management tool provided to administrators.

1.1.1. Server

LENA provides three types of servers: Web Server, Application Server, and Session Server. The purpose of each server is as follows:

1. Web Server: Provides web resources according to user requests. It serves as the front-end for application services provided by Application Server and optionally provides load balancing and security layer (SSL).
2. Application Server: Executes and provides application services written in Java.
3. Session Server: Maintains user sessions between Application Servers.

1.1.2. Agent, Advertiser

Agents installed on Nodes and Servers that handle control and monitoring functions.

- Node Agent
 - Collects Web Server status monitoring data and provides it to Manager.
- Advertiser
 - Collects Application Server status monitoring data and provides it to Manager.

1.1.3. Manager

Manager is a Web Application that provides control and monitoring functions for Nodes and Servers through Node Agent and Advertiser. It typically provides the following functions:

Item	Description
Dashboard	<ul style="list-style-type: none"> • Server, Service Cluster status • Notification confirmation
Server	<ul style="list-style-type: none"> • System (logical Server group) registration/modification/deletion
Service Cluster	<ul style="list-style-type: none"> • Service Cluster registration/modification/deletion • Registered Server list and history inquiry • Service Cluster template and Revision management • CI/CD integration through configuration template download • Remote Terminal and Standard Out/Error Log inquiry (Kubernetes only)
Resource	<ul style="list-style-type: none"> • Resource (Database, DataSource, Application, k8s config) registration/modification/deletion/inquiry • Server list inquiry using resources and addition/removal
Diagnostics (Monitoring)	<ul style="list-style-type: none"> • Issue status monitoring function for Servers • Event inquiry function for Server events
Topology	<ul style="list-style-type: none"> • Server configuration status inquiry by System
Admin	<ul style="list-style-type: none"> • User and permission management, user/permission/menu mapping • User operation history inquiry • License management, status inquiry and upload • Cloud Profile management

1.2. Mechanism

LENA provides monitoring and integrated management functions for Web/Application servers through Manager. However, the environmental difference from the existing Host/VM approach is that Servers running in Containers are controlled by Orchestration tools and do not maintain state.

Therefore, instead of the real-time control/configuration management method through Agents in the existing Host/VM environment, individual Servers download configuration information and licenses through Manager at Container startup time to provide configuration information, and manage by monitoring the status of started Servers through Manager.

As components other than Server, each Server has `docker-entrypoint.sh` which is used as Container startup Command to use download and initial configuration functions at startup, and to transmit the status of operating Servers, Node Agent is installed on Web servers and Application and Session Servers utilize built-in modules.

There are also differences from the existing VM/Host environment in the WEB-WAS integration part. In an environment where IP or addresses are variable due to Container creation/destruction, a Load-balancer is required to maintain continuous connection with Back-End Application Containers, which is

provided in the form of Kubernetes Service, ECS Service Discovery, or EKS/ECS ELB. The method where WEB-WAS were directly connected and WEB servers directly performed Load-balancing in the existing VM/Host environment is not provided in Container environment, and Reverse Proxy connection is provided to Service addresses (Service Endpoints) provided by the platform.

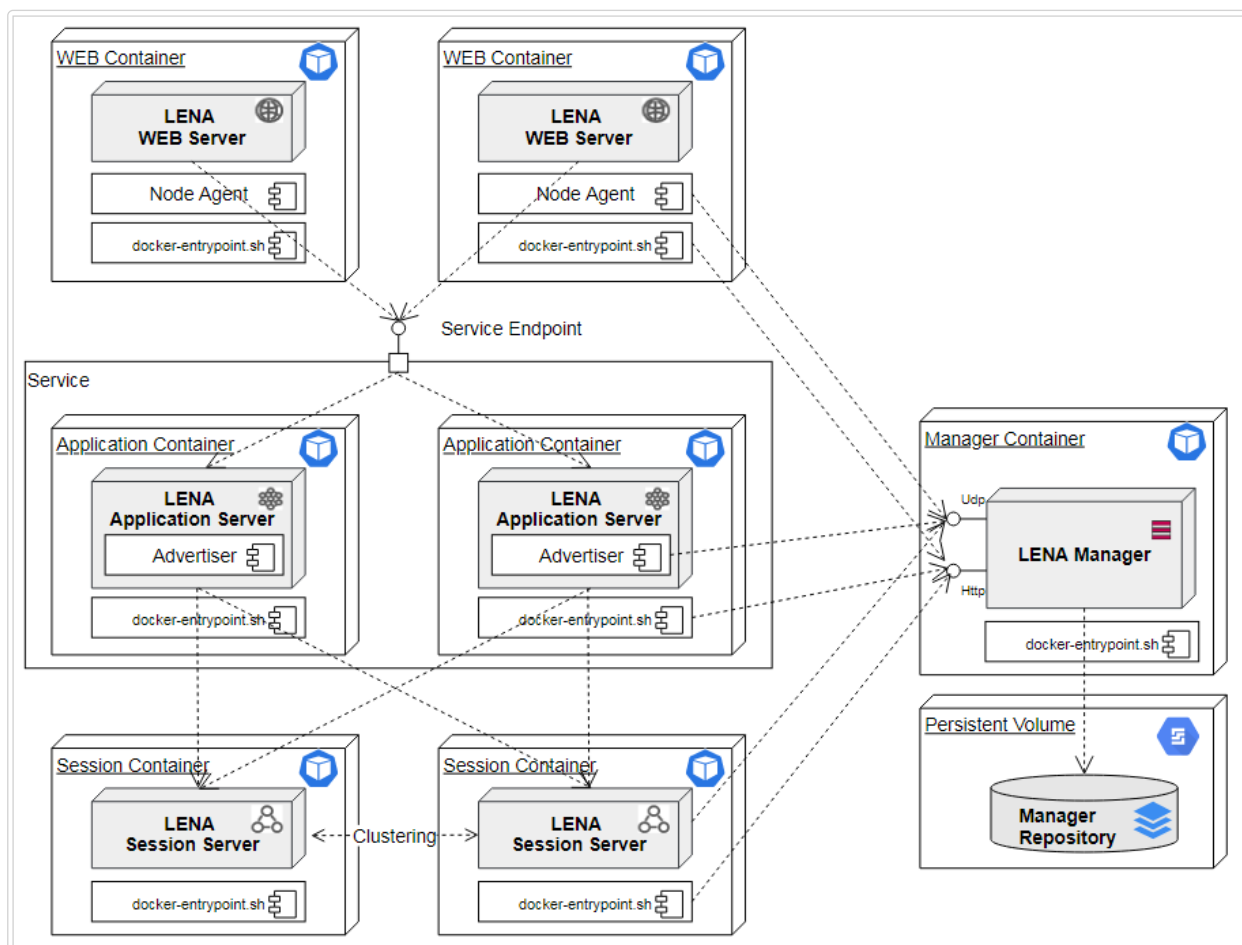


Figure 1. Container-based LENA Server Integration Structure (Kubernetes Environment)

Item	Description	Notes
Application Server	Application Server Instance	
Web Server	Web Server Instance	
Session Server	Session Server Instance	
Manager	Provides configuration file management and Server monitoring functions deployed to servers	
Manager Repository	File storage Repository for Manager operation, includes various configuration information and DB information	Can be separated to external storage

Item	Description	Notes
docker-entrpoint.sh	Shell Script executed when Container starts 1. Initialize configuration information at startup 2. Download configuration information/license from Manager 3. Perform server startup functions	
Node Agent	Collects Web server monitoring data and transmits to Manager, executes control/configuration commands received from Manager	
Advertiser	Collects monitoring data and transmits to Manager	Integrated into Application Server

Depending on the characteristics or constraints of the Container operating environment, Session Server or Manager can also be operated in VM/Host environment, and since LENA Manager includes functions to manage not only Container-type LENA Servers but also VM/Host-based LENA Servers, it can also be operated with the following architecture.

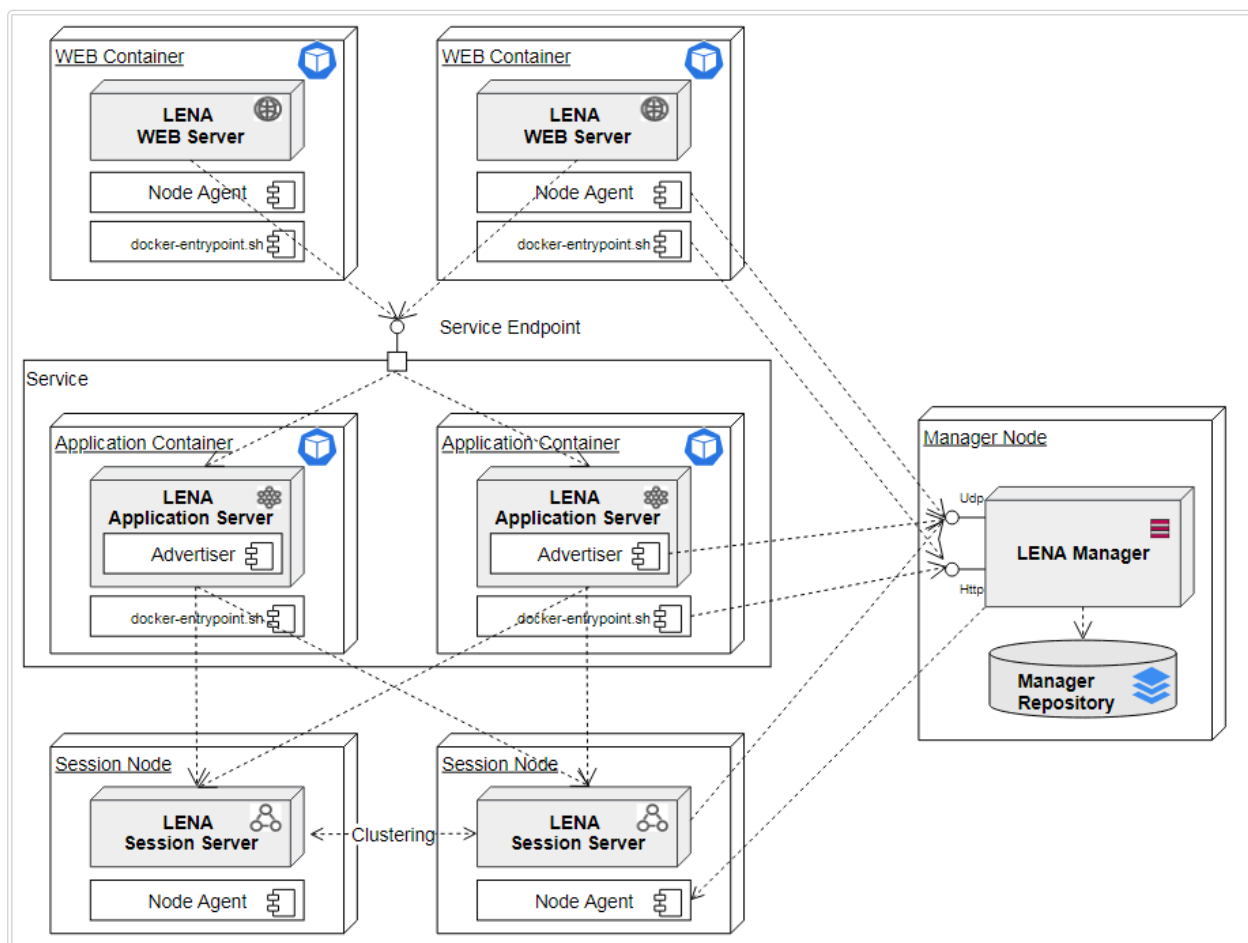


Figure 2. LENA Server Integration Structure in Container-VM/Host Mixed Environment

1.3. Provided Assets

LENA for Container version provides the following assets:

- Docker Image: Images containing Linux OS + JDK + LENA Server + required Libraries provided through Docker Hub
 - Web Server: <https://hub.docker.com/r/lenacloud/lena-web>
 - Application Server: <https://hub.docker.com/r/lenacloud/lena-cluster>
 - Session Server: <https://hub.docker.com/r/lenacloud/lena-session>
 - Manager Server: <https://hub.docker.com/r/lenacloud/lena-manager>
- Kubernetes Manifest files: LENA Server deployment files describing Workload / Service / Config Map required for Kubernetes installation

The specifications of Images provided by Docker Hub are as follows:

Type	JVM	OS (Base Image)	Default Heap Memory
Application Server	Open JDK 1.8	• Cent OS 7 (centos:7)	1.0 GB
Web Server	Open JDK 1.8	• Cent OS 7 (centos:7)	64MB~256MB(Agent)
Session Server	Open JDK 1.8	• Cent OS 7 (centos:7)	1.0 GB
Manager	Open JDK 1.8	• Cent OS 7 (centos:7)	1.0 GB

1.4. System Requirements

The minimum requirements for installing each server instance of LENA for Container are as follows:

Type	JVM	Minimum Memory	Image Size (excluding Base Img)	Default Memory Setting
Application Server	JDK 1.8	512M	Approx. 900 MB (approx. 300MB)	1.25 GB
Web Server	JDK 1.8	512M	Approx. 820 MB (approx. 300MB)	-
Session Server	JDK 1.8	512M	Approx. 900 MB (approx. 300MB)	1.25 GB
Manager	JDK 1.8	512M	Approx. 1,000 MB (approx. 500MB)	1.25 GB

Each server is installed based on the default required Memory during installation, and configuration value changes are required when changing minimum specifications. Image Size is the size of the Image with the entire OS + JDK + LENA Server + required Libraries installed.

Chapter 2. Architecture Decisions

2.1. Process

The overall Architecture decision-making and installation process is as shown in the diagram below.

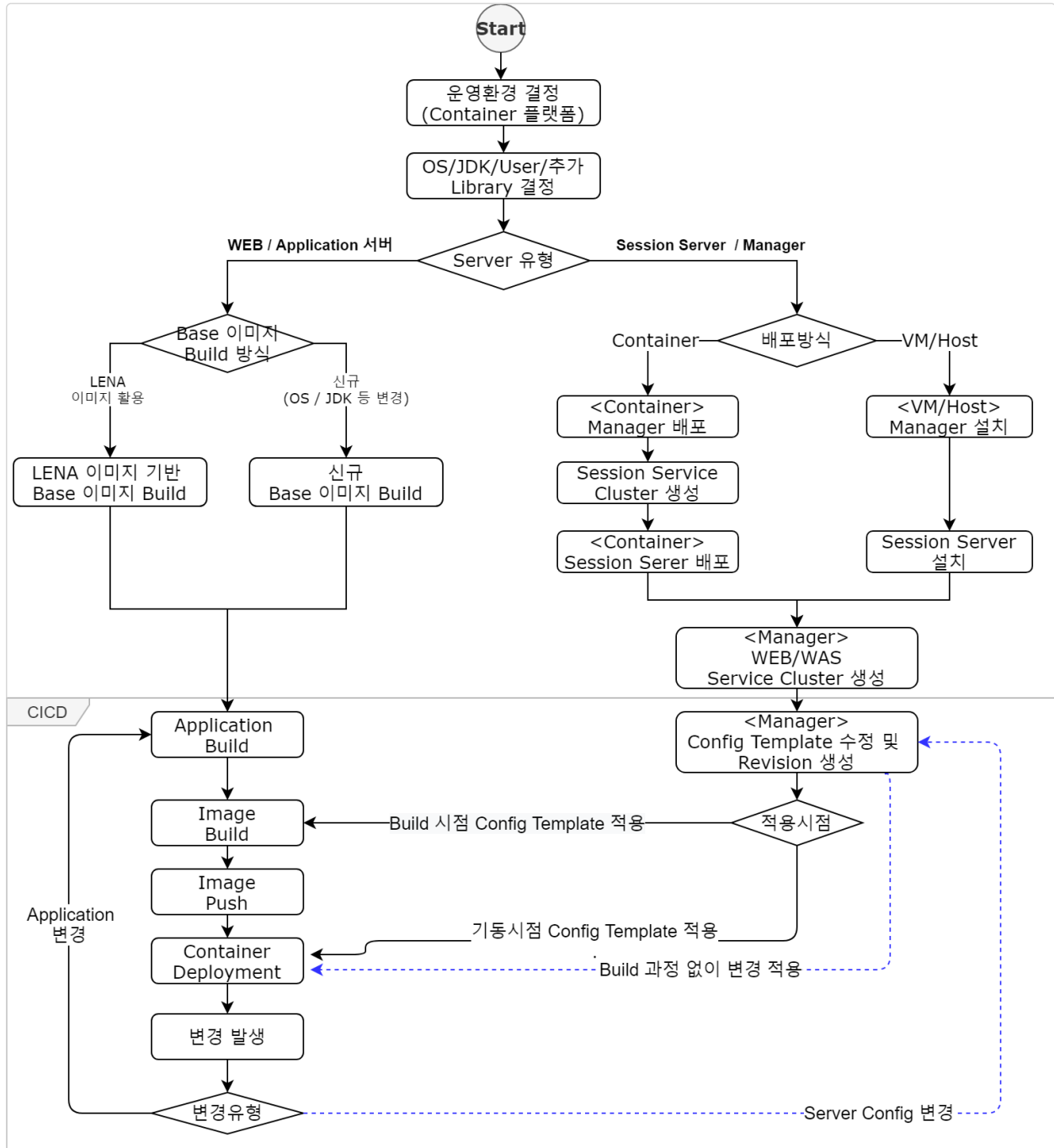


Figure 3. Overall Architecture Decision-making and Installation Process

- The start of Architecture decision-making is determining the Container platform and the OS and JDK of the operating Container. Accordingly, LENA-based Images are selected and deployment methods according to the platform are determined.
- Installation proceeds differently for WEB/WAS servers and Manager/Session servers. Generally, Manager/Session uses LENA Images as provided, while WEB/WAS proceeds by building and utilizing custom Base Images that add Applications/Libraries required for each project based on LENA images.

- To integrate and manage WEB/WAS/Session servers, Service Clusters for each Service must be configured in advance using LENA Manager before configuring the corresponding Containers. When Service Clusters are created and Manager address and Service Cluster information are added to Container environment settings, Template/License download procedures are performed during Container startup, and after startup, Servers of started Containers are automatically registered in Manager and monitoring information can be confirmed.

2.2. Container Operating Environment Considerations

Various Container operating environments are provided by different providers, but they can be broadly divided into two types: Kubernetes environments including EKS/GKE/AKS and Docker-based environments such as Amazon ECS.

Among the characteristics of each Container environment, the key characteristics that affect LENA operations are as follows.

1. Network Communication Between Component Servers

This consideration addresses the ability for mutual communication when servers constituting the system are distributed across internal and external networks of the Container operating environment. Generally, outbound communication is possible unless special network restrictions are set, and for LENA, communication from Server → Manager and WAS → Session Server must be available. Particularly when configuring a mixed VM Container setup, network configuration that enables communication between Containers and VMs is required, similar to ECS VPC network mode.

2. Load Balancing Support

Unlike VM/Host, Containers can be created/destroyed frequently, resulting in IP address changes. Therefore, to maintain continuous service even after back-end Container creation/destruction, a Load-balancer is required in front of back-end services. Kubernetes provides Services that offer Load-Balancing, while ECS provides load-balancing functionality through ELB connections or Service Discovery configuration.

3. Instance Persistence Support (Related Components: Session Server, Manager)

This refers to continuously operating a fixed number of Containers with fixed addresses, which is a necessary characteristic when providing shared resource services like DBMS as Containers. This is required for the configuration of Manager and Session Server among LENA components. Kubernetes provides this characteristic through StatefulSet deployment method, while ECS can operate similarly by deploying Services with Replica 1.

4. External Volume Connection (Related Component: Manager)

Since Containers cannot guarantee state maintenance, information must be stored in external storage (Volume) to continuously maintain data used even when destruction/startup occurs. This is commonly utilized for DBMS database storage or when deploying the same Application to multiple Containers. When operating LENA Manager in Container mode, external Volume connection is required to maintain consistency of management information even after restart.

2.2.1. Container Platform Operating Environment Characteristics

This section examines the characteristics of Container platforms related to the operating environment considerations mentioned above.

Kubernetes

Kubernetes is a portable and scalable Container Orchestration tool for managing containerized Workloads and Services. Physically, Kubernetes is installed and operated in Cluster units consisting of Control Planes and Worker Nodes for Container management. Namespaces, which are logical workspaces, are distributed across Worker Nodes. Kubernetes services consist of Pods (the minimum unit capable of Container deployment), Workloads (units that group and manage Pods), and Services (which provide Workloads as Network services). Workloads and Services are deployed in Namespaces.

Kubernetes Network structure has Cluster Network configured for Service modules, providing Port opening to Host and Load Balancing through this.

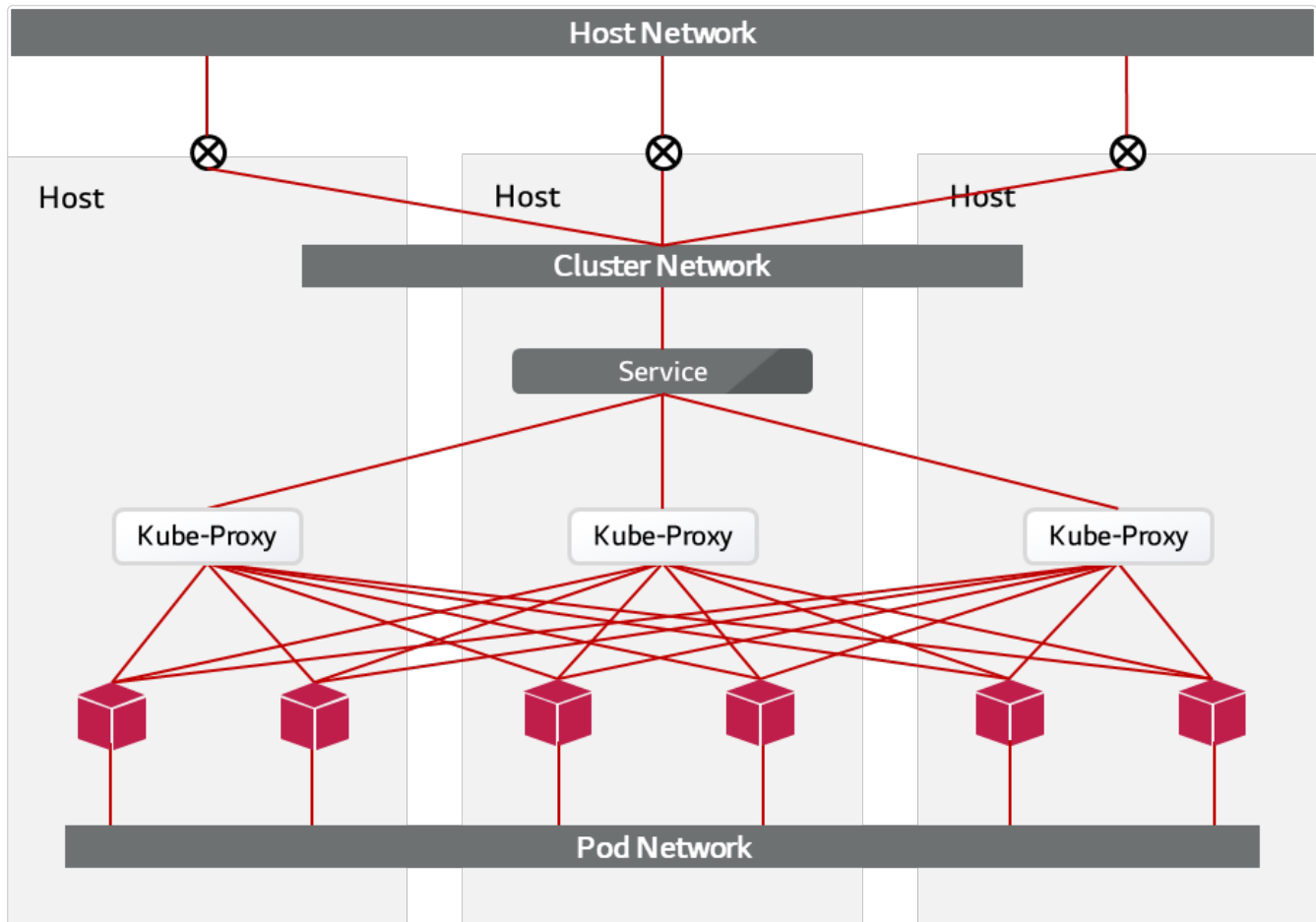


Figure 4. Kubernetes Cluster N/W

Kubernetes Service is an abstracted method for exposing applications running in Pod collections as network services, providing unique IP addresses to Pods and single DNS names for Pod collections while offering Load-Balancing. There are four types of Kubernetes Services as follows.

Cluster IP

Internal fixed IP / Domain Name is assigned in Kubernetes Network, and Cluster Load Balancing is performed through this.

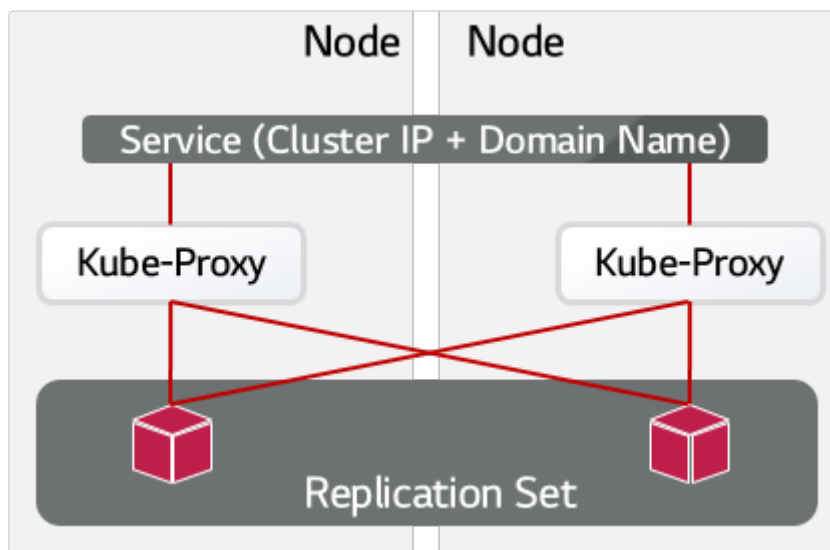


Figure 5. Kubernetes Deployment Type - Cluster IP

Node Port

Connects Ports of all Nodes constituting the Cluster to Container Ports. Ports in the range 30000-32767 are opened externally to Nodes, and while Ports are randomly assigned by default, they can also be assigned as fixed.

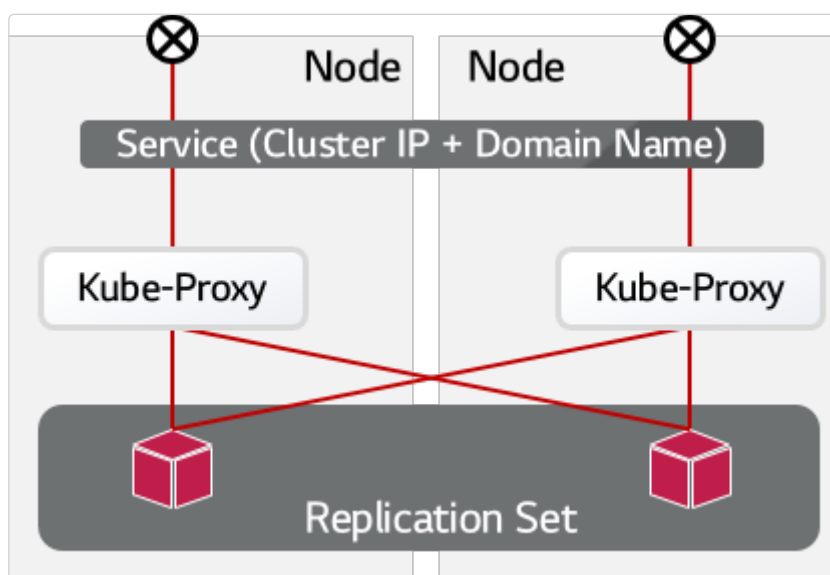


Figure 6. Kubernetes Deployment Type - Node Port

Load Balancer

Opens Node Port while connecting with LoadBalancer outside Container Network to expose Service. In Cloud Services like EKS, Load Balancers provided by Cloud Service Providers are created and connected.

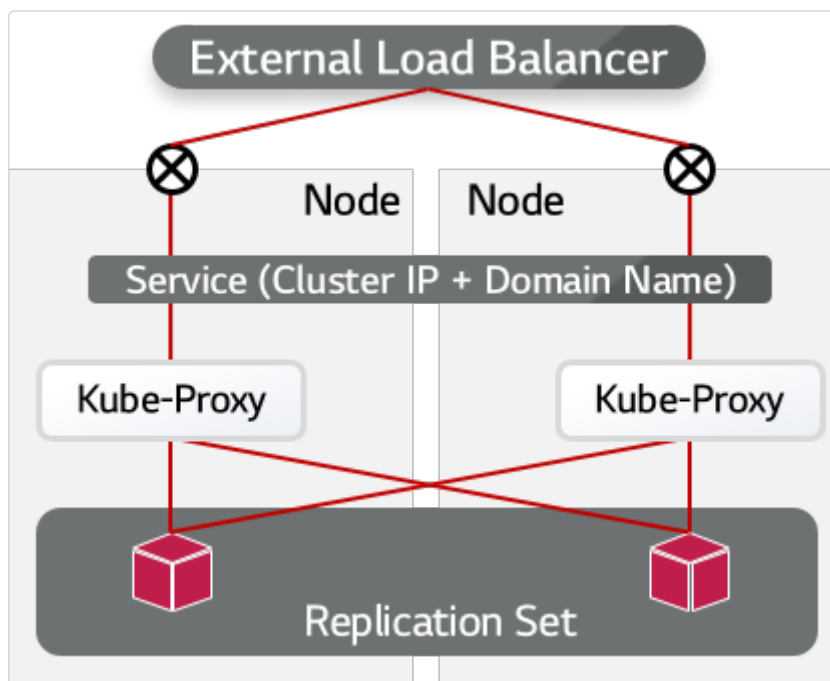


Figure 7. Kubernetes Deployment Type - Load Balancer

Headless

Performs Load Balancing through Domain Name only without a separate Service Cluster IP. Each Pod is assigned its own Domain and is mainly used when utilizing Stateful Sets.

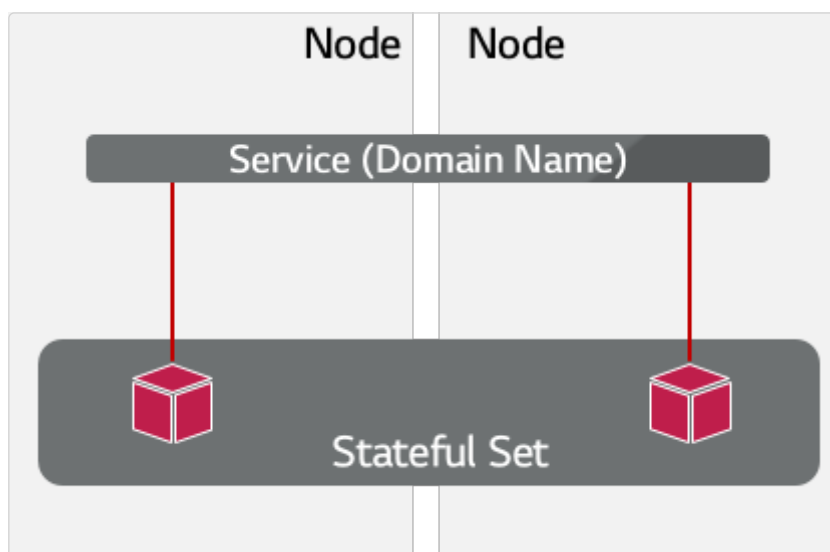


Figure 8. Kubernetes Deployment Type - Headless

Kubernetes supports various types of Container(Pod) deployment methods. While Deployment(Replica Set) is generally used, Stateful Set application is appropriate for LENA Manager and Session Server that require a fixed number of Instances.

Replica Set

Creates the requested number of Replicas regardless of Node count. All Pods can be configured to share the same Persistent Volume.

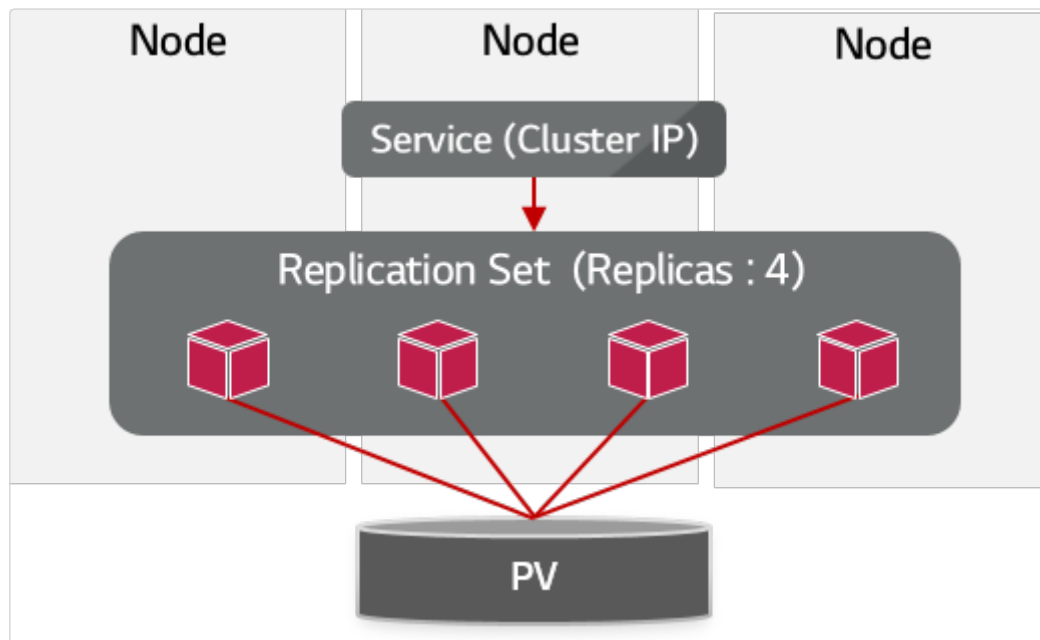


Figure 9. Kubernetes Workload Type - Replica Set

Deployment

Can recreate Replica Sets and perform Versioning. Generally used when deploying WEB servers and Application servers.

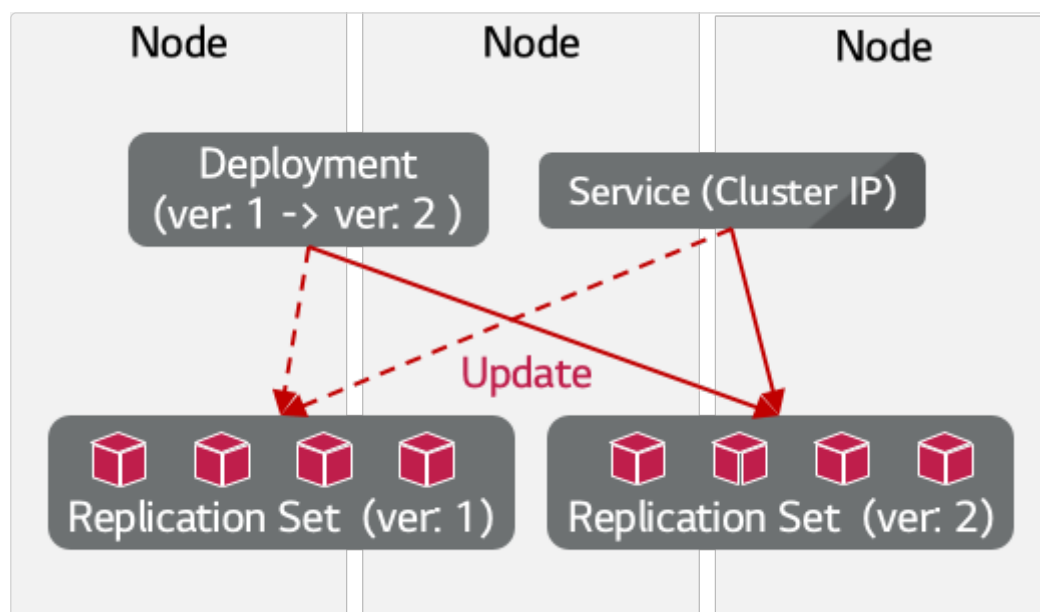


Figure 10. Kubernetes Workload Type - Deployment

Stateful Set

Can maintain a fixed number of Pods, hold state values such as Master/Slave for each Pod, and allocate individual Persistent Volumes per Pod. Generally used when deploying services requiring persistence such as DBMS and Session Servers.

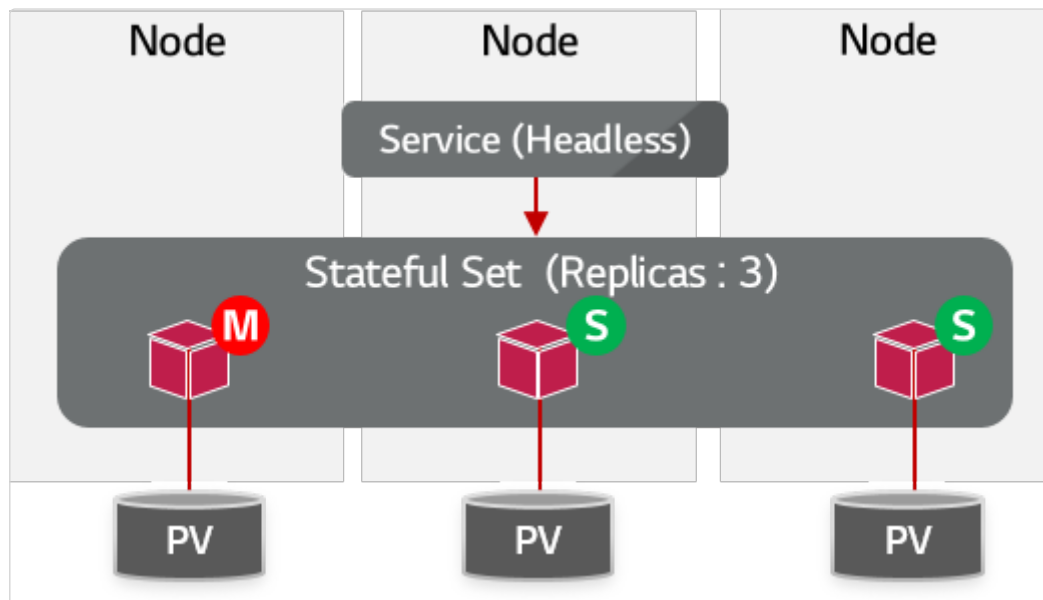


Figure 11. Kubernetes Workload Type - Stateful Set

Daemon Set

Pods equal to the number of Worker Nodes in Kubernetes Cluster are deployed and maintained per Node. Generally used for collecting logs output to Standard Out, collecting monitoring information per Node, or deploying Web servers that replace Ingress.

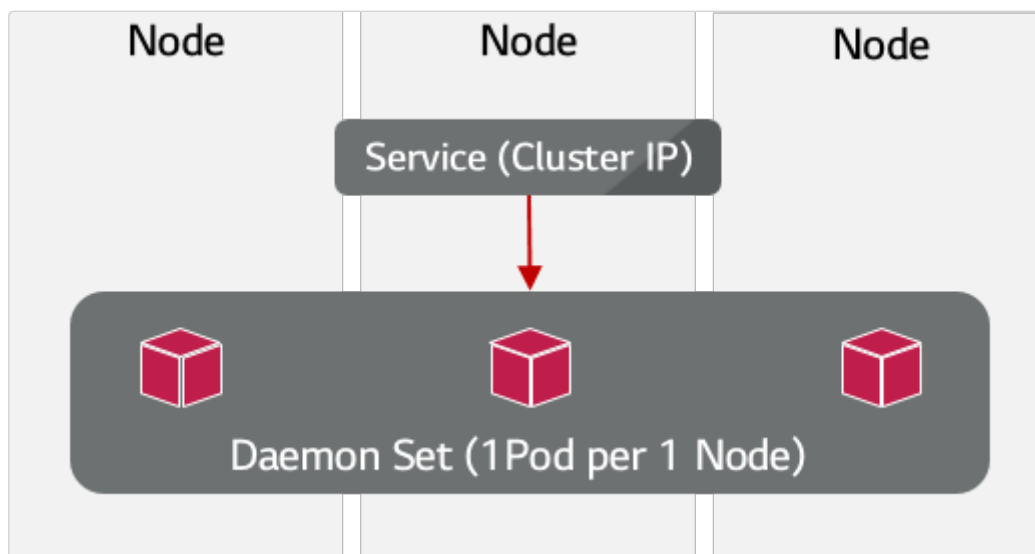


Figure 12. Kubernetes Workload Type - Daemon Set

ECS

AWS ECS consists of Tasks (similar to Kubernetes Pods) and Services that can configure Network, Replica Sets, etc. Load Balancing for Task Instances comprising ECS Services can be provided through 1) ELB method or 2) Service Discovery method. The ELB method can be configured by specifying ELB in Service definition. The Service Discovery method automatically registers Task Instances of ECS services to Amazon Route 53 with DNS names set in Services when created, providing Load Balancing using this. Even when services scale up or down due to external traffic load and container state, Route 53 hosting zones are maintained up-to-date, so connections are made via DNS based on each service's state within the VPC. Route 53 creates A records per Namespace and Task IP, and SRV records per task IP + port to connect to Services.

ECS Key Components

- **Namespace** – Namespace specifies target domain names for routing traffic (e.g., internal, local,

corp). Namespace is a logical boundary between services that must be implemented to be mutually discoverable.

- **Service** – Service is a set of applications contained within a namespace. Services include service instances (Tasks).
- **Task** – An object similar to Kubernetes Pod that groups and manages single or multiple Containers as one Instance, allowing configuration of Container Instance Image/environment settings/Entry Point, etc.

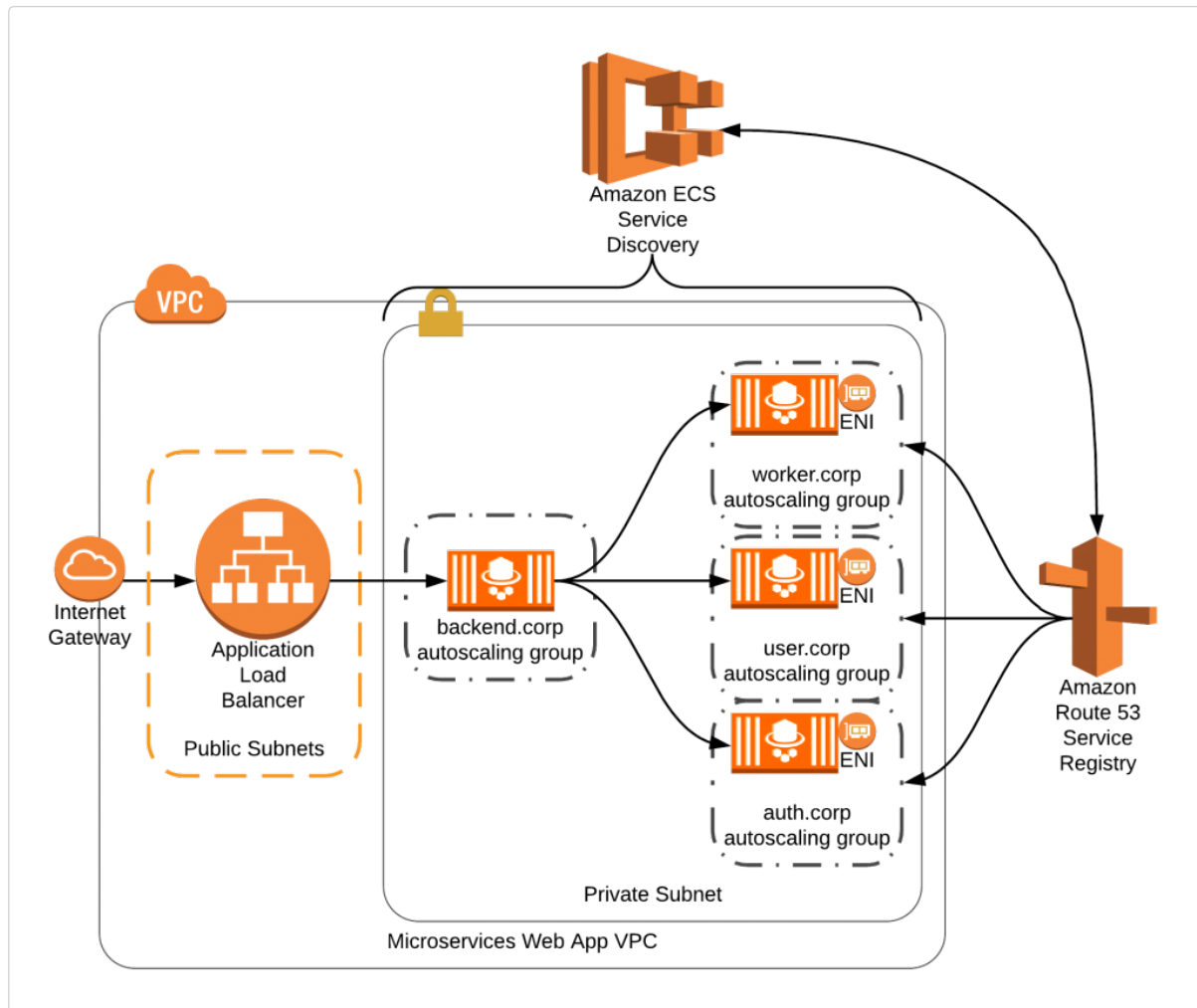


Figure 13. ECS Service Discovery Concept Diagram

Container Platform Characteristics Summary

The characteristics of each environment related to LENA's operating methods are summarized as follows.

Container Operating Environment		External N/W Communication	L/B Support	Instance Persistence	Fixed Address	External Volume Connection
Kubernetes-based	General	Available	Service L/B Support	Supported	Supported	Supported
	EKS	VPC Internal Communication	Service, ELB Connection	Supported	Supported	Supported

Container Operating Environment		External N/W Communication	L/B Support	Instance Persistence	Fixed Address	External Volume Connection
Docker-based	ECS	VPC Internal Communication (VPC N/W Mode)	ELB, Service Discovery Support	Supported(Service Replica=1)	Supported(Service Discovery)	Supported(EFS)

2.2.2. LENA Server Type Operating Methods

Based on the above characteristics, the supported operating methods for Container environments and LENA Server types are as follows.

Container Operating Environment		WEB Server	WAS	Session Server	Manager
Kubernetes-based	General	Container	Container	Container (statefulset), VM/Host	Container (statefulset), VM/Host
Docker-based	ECS	Container	Container	VM, Container	VM, Container

2.3. Common Considerations

The following are elements that must be commonly considered regardless of Server type.

2.3.1. OS

The following OS is used based on LENA Image. This is the Base Image used when building LENA Images.

LENA Image Provided OS	LENA Image Base Image
Cent OS 7	centos:7



When using other OS such as CentOS 8, Ubuntu, Debian, etc., Base Images must be regenerated through LENA technical support.

2.3.2. JDK

Based on LENA Image, OS default JDK 1.8 (installed via yum / apt-get) or Adopt Open JDK 1.8 is installed and used. Installation packages and environment variables are as follows.

OS	JDK Description
Cent OS 7	<ul style="list-style-type: none"> Installation Package: java-1.8.0-openjdk-devel.x86_64 JAVA_HOME: /usr/lib/jvm/java

When using other JDKs, installation is possible during Project Base Image creation, but it is

recommended to install JAVA_HOME Path as similar to the existing one as possible.



Since \$JAVA_HOME environment variable values are already stored in each Server's env.sh (env-manager.sh for manager) and LENA installation information (\${LENA_HOME}/etc/info/java-home.info), when reinstalling JDK, modification of existing file information to match \$JAVA_HOME is required.

2.3.3. Execution User

The execution User based on LENA Image is 'root'. If you want to change this, you must modify the LENA Image and request LENA technical support for the change.

2.3.4. Library

The Libraries installed in LENA Images are as follows.

Library	Purpose	OS Application	Applied Server
net-tools	N/W utilities such as wget	(Common)	(Common)
hostname	For Hostname verification	CentOS	(Common)
initscript	For Service (Daemon) operation	CentOS	(Common)
procps	Process-related utilities	CentOS	(Common)
unzip	For Server configuration file decompression	(Common)	(Common)
file	For File format verification	(Common)	(Common)
curl	For file download	Ubuntu / Debian	(Common)
cronie-noanacron	For Crontab operation	CentOS	(Common)
logrotate	For WEB/WAS File Log Rotate processing	(Common)	(Common)
libxml2-utils	For XML Validation (License file Validation)	Debian	(Common)
locales	For Locale configuration	Ubuntu / Debian	(Common)
libapr1	Web Server Library	Ubuntu / Debian	(Common)
libaprutil1	Web Server Library	Ubuntu / Debian	(Common)
tzdata	Time Zone configuration	Ubuntu / Debian	(Common)
openssl	Web / WAS Library	(Common)	(Common)

Library	Purpose	OS Application	Applied Server
awscli pip	For EKS API calls from Manager	(Common)	Manager

2.4. Server Type Considerations – Manager

2.4.1. Deployment

Manager deployment is possible in both Container or VM/Host deployment, and the constraints for applying both methods are as follows.

1. Fixed Domain or IP address allocation
Since Servers installed in Containers download configuration files/licenses and transmit monitoring information, they must be served with fixed addresses even after reboot/regeneration for continuous service.
2. Server → Manager N/W communication
Unidirectional communication is required for using Manager download services and providing monitoring information. Based on default settings, TCP Port 7700, UDP/TCP Port 16100 access must be allowed
3. Persistent Volume
For Managers installed in Containers, external Volumes capable of storing DB, configuration information, monitoring data, etc. are required to provide service continuity even after restart. Secure Persistent Volumes using NFS, EBS Disk, Local Node Disk, etc., and allocate to Manager Container for use
4. Instance persistence guarantee
Manager Instances must be guaranteed Instance persistence to provide service continuity. While general VM/Host basically guarantees persistence, in Container environments they must be deployed in ways that guarantee persistence like Kubernetes StatefulSet

Deployment Method	Constraints (Requirements)
Container Deployment	<ul style="list-style-type: none"> • Persistent Volume • Server → Manager N/W communication (TCP Port 7700, UDP/TCP Port 16100) • Fixed domain or fixed IP allocation • Container persistence guarantee
VM/Host Installation	<ul style="list-style-type: none"> • Server → Manager N/W communication (TCP Port 7700, UDP/TCP Port 16100) • Fixed domain or fixed IP allocation

2.4.2. Specifications

The specifications required to operate Manager Server are as follows.

Memory

Manager Server requires a minimum Heap Memory Size of 512Mbyte and the Image has the following default settings.

- Heap Memory : 1024 Mbyte
- Metaspace Memory : 256 Mbyte

To change these, adjust by setting the following environment variables.

- LENA_JVM_HEAP_SIZE
- LENA_JVM_METASPACE_SIZE



The values of the above environment variables are in MByte units and must be specified in the format of number+'m' (e.g., 1024m). If the format is inconsistent, it will not be applied.

Disk

Based on LENA Image, the Image size used is approximately 1,000 Mbyte, which is the total of OS + JDK + LENA + Library including the capacity of Image upper layers.

Additional disk capacity to consider includes: 1) Manager Log file capacity and 2) Repository (DB and file storage). The Repository location used by Manager is `${LENA_HOME}/repository` and requires approximately 5GB of capacity. When installed in Container, this Repository should be connected to Persistent Volume and stored outside the Container.



The Repository location used by Manager is `${LENA_HOME}/repository`. When installed in Container, it is recommended to connect to Persistent Volume and store outside the Container for data persistence even after Container restart.

2.4.3. Configuration

Network

1. Network Address

Manager must be assigned a fixed Domain or IP address. (Refer to [Deployment](#) section in this document)

2. Service Port

Manager service ports are fixed as follows:

- Http (TCP) Port 7700: Integrated management service and Rest API service provision
- UDP Port 16100: Monitoring data collection
- TCP Port 16100: Thread / Service Dump data generation / collection

Ports are fixed in LENA Manager Image, and when changes are desired, it is recommended to change Port Mapping according to Container's basic configuration philosophy rather than changing LENA Image.



Changing LENA Manager's Service port means changing integration ports with other Servers, and when changed, it requires configuration changes/restart of all connected Servers.

Environment Variables

The main environment variables applicable to Manager Container are as follows.

Environment Variable	Description	Default Value	Changeable
LENA_JVM_HEAP_SIZE	<ul style="list-style-type: none"> Specify Heap Memory size 	1024m	<input type="radio"/>
LENA_JVM_METASPACE_SIZE	<ul style="list-style-type: none"> Metaspace Memory size 	256m	<input type="radio"/>
LENA_MANAGER_DOMAIN_ENABLED	<ul style="list-style-type: none"> Domain Name activation status 'Y' or 'N' 	Y	<input type="radio"/>
LENA_MANAGER_ADDRESS	<ul style="list-style-type: none"> Manager address exposed externally (recognized by Servers) Format: IP / Domain address : service port Example) For Kubernetes: Service Domain 		<input type="radio"/>
JAVA_DOMAIN_CACHE_TTL	<ul style="list-style-type: none"> Domain address Cache time (seconds) 	3	<input type="radio"/>
LENA_SERVER_TYPE	<ul style="list-style-type: none"> Server type 	manager	<input checked="" type="radio"/>
LENA_HOME	<ul style="list-style-type: none"> LENA installation location 	/usr/local/lena	<input checked="" type="radio"/>
LENA_JVM_OPTIONS	<ul style="list-style-type: none"> User-defined JVM OPTION 		<input type="radio"/>
LENA_USER	<ul style="list-style-type: none"> OS user account to use for Manager startup 	root	<input type="radio"/>
LENA_USER_GROUP	<ul style="list-style-type: none"> OS user group to use for Manager startup 	root	<input type="radio"/>



When JAVA_DOMAIN_CACHE_TTL value is set, it changes the networkaddress.cache.ttl value in \${JAVA_HOME}/jre/lib/security/java.security file.

Directory Structure

Based on LENA Image, the default installation location is '/usr/local/lena', and its subdirectory structure is as follows.

Directory (Under \${LENA_HOME})	Description	Notes
bin	Manager's Start/Stop scripts	

Directory (Under \${LENA_HOME})	Description	Notes
depot	Local Repository for installation	
etc	Other meta information and configuration files	
license	Directory for managing License information	
logs └ lena-manager	Log file storage Home Manager Log file storage	
modules └ lena-manager	Storage Home for LENA-provided modules Path where modules required for lena-manager execution are located	
repository └ backup └ config └ container └ database └ license └ monitoringDB └ resource └ template	Manager data storage Home Backup data storage Manager configuration information storage Container configuration information storage Manager database storage License file storage Monitoring data storage Resource upload file storage Server configuration Template storage	Recommend using external volume in Container
tmp	Temporary directory	

Log & Dump Output

Log and Dump support both 'console' method outputting to Standard Out / Error and 'file' method outputting to File, and the output method can be switched by setting the environment variable 'LOG_OUTPUT_TYPE' value to 'console' or 'file'.

1. Console Output

This is a method commonly used in Container environments. Manager's Application Log, Access Log, and GC Log are all output to Standard Out. They are stored at designated locations on Node(Host) by Log Driver set in Docker (default location: /var/lib/docker/containers/[container-id]/[container-id]-json.log) or can be collected/stored by Log Aggregator such as FluentD for integrated management.

2. File Output

When file output is set, Log files and Dump files are stored under \${LENA_HOME}/logs/lena-manager in Daily Rolling method. According to default settings, stored files delete Log files that have passed more than 30 days since last recorded, daily.

This method can be used when Manager operates in VM/Host environment or in environments where Manager Logs are not separately collected.

Health Check

Health Check is explained based on Kubernetes.

Health Check includes 1) Readiness Probe that determines whether Container service is ready during startup, 2) Liveness Probe that checks whether service is operating normally during operation, and 3) Startup Probe that determines whether Application has started.

Category	Purpose
Readiness (Probe)	At startup time, whether container is ready to process requests
Liveness (Probe)	During operation, whether container is operating normally
Startup (Probe)	Whether application within container has started

There are the following 3 types of Check methods (Action).

Category (Action)	Method
TCP Socket (TCPSocketAction)	Health Check by Port communication status
Http URL Query (HTTPGetAction)	Health Check by URL call result code
Exec Execution (ExecAction)	Health Check by executing commands inside Container

The method for Health Check of Manager uses URL check method, applying Readiness and Liveness Probe. The default settings are as follows.

1. Readiness Check

- httpGet : path /lena, port 7700
- initialDelaySeconds : 5
- periodSeconds : 5

2. Liveness Check

- httpGet : path /lena, port 7700
- initialDelaySeconds : 20
- periodSeconds : 5

2.5. Server Type Considerations – Session Server

2.5.1. Deployment

Session Server deployment is possible in both Container or VM/Host deployment, and the constraints for applying both methods are as follows.

1. Fixed Domain or IP address allocation

Session Server is configured as Cluster with 2 Containers. Each Session Server recognizes the Domain/IP of the counterpart Session Server as Mirror Server information and synchronizes Session information, so for continuous service, it must be served with fixed addresses even after reboot/regeneration.

2. Instance persistence guarantee

Session Instances must be guaranteed Instance persistence to provide service continuity. General VM/Host basically guarantees persistence, but in Container environments they must be deployed in ways that guarantee persistence like Kubernetes StatefulSet.

Deployment Method	Constraints (Requirements)
Container Deployment	<ul style="list-style-type: none">• Fixed domain or fixed IP allocation• Container persistence guarantee
VM/Host Installation	<ul style="list-style-type: none">• Fixed domain or fixed IP allocation

2.5.2. Specifications

The specifications required to operate Session Server are as follows.

Memory

Session Server requires a minimum Heap Memory Size of 1024Mbyte and the Image has the following default settings.

- Heap Memory : 1024 Mbyte

To change this, adjust by setting the following environment variable.

- LENA_JVM_HEAP_SIZE



The value of the above environment variable is in MByte units and must be specified in the format of number+'m' (e.g., 1024m). If the format is inconsistent, it will not be applied.

Disk

Based on LENA Image, the Image size used is approximately 800 Mbyte, which is the total of OS + JDK + LENA + Library including the capacity of Image upper layers.

Additional disk capacity to consider here is Session Server Log files.

Configuration

Network

1. Network Address

Session Server must be assigned a fixed Domain or IP address. (Refer to [Deployment Method](#) section in this document) Session information is communicated with fixed addresses from WAS and Secondary Session Server.

2. Service Port

Session Server service ports are fixed as follows:

- Http (TCP) Port 5180* : Session inquiry and management port

The above ports are fixed in LENA Image, and when changes are desired, it is recommended to change Port Mapping according to Container's basic configuration philosophy rather than changing LENA Image.



Changing Session Server's Service port means changing integration ports with other Servers, and when changed, it requires configuration changes/restart of all connected Servers.

Environment Variables

The main environment variables applicable to Session Container are as follows.

Environment Variable	Description	Default Value	Changeable
LENA_JVM_HEAP_SIZE	<ul style="list-style-type: none"> Specify Heap Memory size 	1024m	<input type="radio"/>
LENA_MANAGER_ADDRESS	<ul style="list-style-type: none"> Manager address exposed externally (recognized by Servers) Format: IP / Domain address : service port Example) For Kubernetes: Service Domain 		<input type="radio"/>
LENA_CONFIG_TEMPLATE_ID	<ul style="list-style-type: none"> Service Cluster name : Revision No 		<input type="radio"/>
JAVA_DOMAIN_CACHE_TTL	<ul style="list-style-type: none"> Domain address Cache time (seconds) 	0	<input type="radio"/>
LENA_SESSION_0_ADDRESS	<ul style="list-style-type: none"> Primary Session server address, must match StatefulSet configuration 		<input type="radio"/>
LENA_SESSION_1_ADDRESS	<ul style="list-style-type: none"> Secondary Session server address, must match StatefulSet configuration 		<input type="radio"/>
LENA_SECONDARY_SESSION_NO	<ul style="list-style-type: none"> Select information to use as mirror server from LENA_SESSION_0_ADDRESS / LENA_SESSION_1_ADDRESS (only 0, 1 input allowed) 		<input type="radio"/>
LENA_SESSION_EXPIRE_SEC	<ul style="list-style-type: none"> Session expiration time (seconds) 	1800	<input type="radio"/>
LENA_CONFIG_SHARE_SESSION	<ul style="list-style-type: none"> Session sharing between Applications 'Y' or 'N' values allowed 	N	<input type="radio"/>
LENA_SERVER_TYPE	<ul style="list-style-type: none"> Server type 	session	X
LENA_HOME	<ul style="list-style-type: none"> LENA installation location Default value: /usr/local/lena 	(See description)	X

Environment Variable	Description	Default Value	Changeable
LENA_SERVER_HOME	<ul style="list-style-type: none"> Session Server installation location Default value: /usr/local/lena/server/sessionServer 	(See description)	X
LOG_OUTPUT_TYPE	<ul style="list-style-type: none"> Log output method (file/console) 	console	○
LENA_AGENT_RUN	<ul style="list-style-type: none"> LENA Agent startup status 	N	○
LENA_USER	<ul style="list-style-type: none"> OS user account to use for Manager startup 	root	○
LENA_USER_GROUP	<ul style="list-style-type: none"> OS user group to use for Manager startup 	root	○



When JAVA_DOMAIN_CACHE_TTL is set, it changes the networkaddress.cache.ttl value in \${JAVA_HOME}/jre/lib/security/java.security file.

Directory Structure

Based on LENA Image, the default installation location is '/usr/local/lena', and its subdirectory structure is as follows.

Directory (Under \${LENA_HOME})	Description	Notes
bin	Session Server's Start/Stop scripts	Not used
depot	Local Repository for installation	Not used
etc	Other meta information and configuration files	
license	Directory for managing License	
modules	Storage Home for LENA-provided modules	
servers/sessionServer └ lib └ logs	Session Server installation location (\${LENA_SERVER_HOME}) Session Server Library storage Log file storage	
tmp	Temporary directory	

Log

Session Server provides only File Log output. The output location is \${LENA_SERVER_HOME}/logs directory and the file name format is lena-sessionServer-YYYYMMDD.log with Log files created daily.

Health Check

For basic Health Check content, refer to the [Manager Health Check](#) section in this document.

Based on Kubernetes, the method for Health Check of Session Server is Command Exec (ExecAction) method, calling `${LENA_SERVER_HOME}/health.sh`.

1. Readiness Check
 - `exec` : `${LENA_SERVER_HOME}/health.sh`
 - `initialDelaySeconds` : 20
2. Liveness Check
 - `exec` : `${LENA_SERVER_HOME}/health.sh`
 - `initialDelaySeconds` : 30
 - `periodSeconds` : 5

2.6. Server Type Considerations – WAS

2.6.1. Deployment

WAS is deployed in Container, and according to configuration, single or multiple Instances are simultaneously deployed to Container platforms (Kubernetes, ECS, etc.).

1. Service Planning

WAS is deployed as single/multiple Containers, and to service this to external or Front-End, it is a common approach to place a Service that performs L/B role in the front. In the case of Kubernetes, service types are provided including NodePort that serves specific Ports of installed Node, LoadBalancer that utilizes external L/B, and ClusterIp that specifies internal fixed IP, while ECS has a method of specifying ALB. It is necessary to decide in advance how to service Application.
2. Instance Number (Replica)

The number of multiple WAS serving a single Service is variable according to load, but the number of Instances to initially start must be predefined and reflected in deployment configuration.
3. Service Mapping

In the case of ECS, L/B can be directly specified in Service, but in the case of Kubernetes, Mapping rules are defined based on labels defined as Key-Value and mapped with Service. A Mapping standard that is non-duplicative across the entire system and convenient for operation must be established and reflected in deployment configuration.

2.6.2. Specifications

The specifications required to operate WAS are as follows.

Memory

WAS requires a minimum Heap Memory Size of 512Mbyte and the Image has the following default settings.

- Heap Memory : 1024 Mbyte
- Metaspace Memory : 128 Mbyte

To change these, adjust by setting the following environment variables.

- `LENA_JVM_HEAP_SIZE`
- `LENA_JVM_METASPACE_SIZE`



The values of the above environment variables are in MByte units and must be specified in the format of number+'m' (e.g., 1024m). If the format is inconsistent, it will not be applied.

Disk

Based on LENA Image, the Image size used is approximately 900 Mbyte, which is the total of OS + JDK + LENA + Library including the capacity of Image upper layers.

Additional Disk required for operation is calculated considering Log capacity when storing Logs in file method and Application source file (Artifact) capacity.

2.6.3. Configuration

Network

1. Network Address

WAS network address has no special constraints. It must be placed to enable unidirectional communication with Manager and Session Server as destinations on N/W.

2. Service Port

WAS service ports are fixed as follows.

- HTTP service Port : 8180

The above ports are fixed in LENA Image, and when external service changes are desired, it is recommended to change through Port binding configuration according to Container's basic philosophy rather than changing LENA Image.

Environment Variables

The main environment variables applicable to WAS Container are as follows.

Environment Variable	Description	Default Value	Changeable
LENA_SERVICE_PORT	• WAS service Port	8180	○
LENA_JVM_HEAP_SIZE	• Specify Heap Memory size	1024m	○
LENA_JVM_METASPACE_SIZE	• Metaspace Memory size	128m	○
LENA_JVM_OPTIONS	• User-defined JVM OPTION		○
LENA_MANAGER_ADDRESS	• Manager address • Format: IP / Domain address : service port		○
LENA_MANAGER_MONITORING_PORT	• Manager monitoring Port information	16100	○
LENA_MANAGER_KEY	• Manager Open API access token		○

Environment Variable	Description	Default Value	Changeable
LENA_CONFIG_TEMPLATE_DOWNLOAD	<ul style="list-style-type: none"> Whether to download configuration file from Manager Allowed values: Y or N 		<input type="radio"/>
LENA_CONFIG_TEMPLATE_ID	<ul style="list-style-type: none"> Configuration file ID Format: Service Cluster name:Revision number 		<input type="radio"/>
LENA_LICENSE_DOWNLOAD_URL	<ul style="list-style-type: none"> License download URL Input value: manager or customer-owned License download URI 	manager	<input type="radio"/>
LENA_CONTRACT_CODE	<ul style="list-style-type: none"> Encrypted value related to License issuance contract code. 		<input type="radio"/>
JAVA_DOMAIN_CACHE_TTL	<ul style="list-style-type: none"> Domain address Cache time (seconds) 	3	<input type="radio"/>
LOG_OUTPUT_TYPE	<ul style="list-style-type: none"> LOG output type Allowed values: console or file 	console	<input type="radio"/>
LENA_LOG_OUTPUT_DIR	<ul style="list-style-type: none"> Log file creation location 	/usr/local/l ena/server s/appServ er/logs	<input type="radio"/>
LENA_DUMP_OUTPUT_DIR	<ul style="list-style-type: none"> Dump file creation location 		<input type="radio"/>
LENA_SERVER_TYPE	<ul style="list-style-type: none"> Server type 	WAS	X
LENA_HOME	<ul style="list-style-type: none"> LENA installation Home Value: /usr/local/lena 	(See description)	X
LENA_SERVER_HOME	<ul style="list-style-type: none"> LENA server installation location Value: /usr/local/lena/servers/appServer 	(See description)	X
LENA_SERVICE_ENDPOINT	<ul style="list-style-type: none"> Address of service that WAS belongs to 		<input type="radio"/>
LENA_AGENT_RUN	<ul style="list-style-type: none"> LENA Agent startup status 	N	<input type="radio"/>
LENA_USER	<ul style="list-style-type: none"> OS user account to use for Manager startup 	root	<input type="radio"/>

Environment Variable	Description	Default Value	Changeable
LENA_USER_GROUP	• OS user group to use for Manager startup	root	<input type="radio"/>
LENA_HEALTH_CHECK	• Whether to perform Health Check	N	<input type="radio"/>
LENA_HEALTH_CHECK_WAS_URL	• Page information for Health Check	/tie/lenaHealthCheck.jsp	<input type="radio"/>
LENA_HEALTH_CHECK_INITIAL_DELAY_MILLI_SEC	• Wait time before Health Check starts after LENA Agent startup, to secure Server startup time	60000 (milliseconds)	<input type="radio"/>
LENA_HEALTH_CHECK_TIMEOUT_MILLI_SEC	• Health Check request Timeout	5000 (milliseconds)	<input type="radio"/>
LENA_HEALTH_CHECK_FAILURE_THRESHOLD	• Health Check failure threshold	5	<input type="radio"/>
LENA_HEALTH_CHECK_TERM_EXECUTION	• Whether to perform follow-up tasks when Health Check failure threshold is exceeded	true	<input type="radio"/>
LENA_HEALTH_CHECK_TERM_EXECUTION_SCRIPT	• Follow-up task script information when Health Check failure threshold is exceeded	stop-container	<input type="radio"/>
LENA_HEALTH_CHECK_TERM_EXECUTION_INTERVAL	• Follow-up task execution cycle when Health Check failure threshold is exceeded	300 (seconds)	<input type="radio"/>



- LENA_CONFIG_TEMPLATE_ID: Revision number can be omitted, and when omitted, Default Revision is downloaded.
- LENA_CONTRACT_CODE: Used for License validity check, and if this value is invalid, license download is canceled.
- JAVA_DOMAIN_CACHE_TTL: When this value is set, it changes the networkaddress.cache.ttl value in `${JAVA_HOME}/jre/lib/security/java.security` file.
- LOG_OUTPUT_TYPE: When Server configuration file download is applied, Log settings of the downloaded configuration file are applied.

Directory Structure

Based on LENA Image, the default installation location is '/usr/local/lena', and its subdirectory structure is as follows.

Directory (Under <code>\${LENA_HOME}</code>)	Description	Notes
bin	Node Agent's Start/Stop scripts	Not used

Directory (Under <code>\${LENA_HOME}</code>)	Description	Notes
depot	Local Repository for installation	Not used
etc	Other meta information and configuration files	
license	Directory for managing License information	
logs	LENA management log file storage	
modules	Storage Home for LENA-provided modules	
└ lena-agent	Path where modules required for Node Agent execution are located	Not used
servers/webServer	Server installation Home, <code>\${LENA_SERVER_HOME}</code>	
└ bin	Server Start / Stop / management execution Script storage	
└ conf	Server configuration information storage	
└ dumps	Dump file storage	
└ hook	Life-Cycle Hook Shell file storage	
└ lib	Server execution Library storage	
└ logs	Log file storage	
└ temp	Working temporary directory	
└ webapps	Default Application Deployment directory	
└ work	JSP Servlet conversion source and compilation result storage	
tmp	LENA management temporary directory	

Log & Dump Output

Log supports both 'console' method outputting to Standard Out / Error and 'file' method outputting to File, and the output method can be switched by setting the environment variable 'LOG_OUTPUT_TYPE' value to 'console' or 'file'.

1. Console Output

This is a method commonly used in Container environments. Server's Application Log, Access Log, and GC Log are all output to Standard Out. They are stored at designated locations on Node(Host) by Log Driver set in Docker (default Docker Log file location: `/var/lib/docker/containers/[container-id]/[container-id]-json.log`) or can be collected/stored by Log Aggregator such as FluentD for integrated management.

2. File Output

When file output is set, Log files and Dump files are stored as files under `${LENA_HOME}/servers/appServer/logs` directory, and each Log file is daily rolled by logrotate settings.

The types of output Log files and output file names are as shown in the table below.

Log Type	Output Location
Access Log	access_appServer_\${HOSTNAME}.log
GC Log	gc_appServer_\${HOSTNAME}.log
Application Log	appServer_lena-\${HOSTNAME}.out.log

These Log files are rolled daily by logrotate installed on Guest OS.

When outputting file-based logs, it is common to configure a Log management Stack (ELK, EFK Stack, etc.) that collects/queries logs through a separate Log Aggregator. For this purpose, it is common to add Side-car Containers such as Fluent-Bit to collect logs.

Dump files are created at the following location.

- Dump Home : \${LENA_HOME}/servers/appServer/dumps/ \${HOSTNAME}

The \${HOSTNAME} directory is for distinguishing between Containers when storing Dump files in external Volume.

Storage locations by Dump type are as follows.

- Heap Dump : \${DUMP_HOME}/hdump
- Thread Dump : \${DUMP_HOME}/tdump
- Service Dump : \${DUMP_HOME}/sdump

Health Check

For basic Health Check content, refer to the [Manager Health Check](#) section in this document.

Generally, WAS uses Http Get method for Health Check, but LENA WAS's default Health Check method is set to TCP Port check method. This is because basic LENA Image does not have Business Application mounted, and when Biz Application is mounted, it is recommended to update the appropriate Http Get Health Check settings for that Application. The default settings based on provided Kubernetes Manifest files are as follows.

1. Readiness Check

- TCPSocketAction : port 8180
- initialDelaySeconds : xx (adjustment required according to Application characteristics)
- timeoutSeconds : xx (adjustment required according to Application characteristics)

2. Liveness Check

- TCPSocketAction : port 8180
- initialDelaySeconds : xx (adjustment required according to Application characteristics)
- periodSeconds : xx (adjustment required according to Application characteristics)



- In the case of LENA WAS, the Service Port changes to Listen state after normal Server startup.
- Since Health Check Page is judged to provide normal service when Check succeeds, it is recommended to select and apply a Page that can determine whether the service's Back-end (e.g., Database) is also normal.

Server Configuration Management

The procedure for Container and WAS startup is as follows.

Generally, Server configuration must be applied before WAS startup, and the timing for applying configuration includes: 1) including in Base Image, 2) reflecting at Container startup time, and 3) including in Application Artifact.

Configuration Application Method	Description	LENA Function Support
Include in Base Image	Copy configuration information and include in Image when creating Base Image	○
Apply at Startup Time	Copy configuration information from external Repository when Container starts	○
Include in Application Artifact	Like Spring Boot case, Server configuration information is simplified and included in Application Artifact, applied together with Application Artifact deployment	○ Provided through WAS(Embedded)

In LENA, Server configuration information pre-configured through Manager can be reflected at Image Build time or Container startup time. For this purpose, Manager must be installed and Server configuration information must be configured before configuring Application / WEB Server Container. This is illustrated as follows.

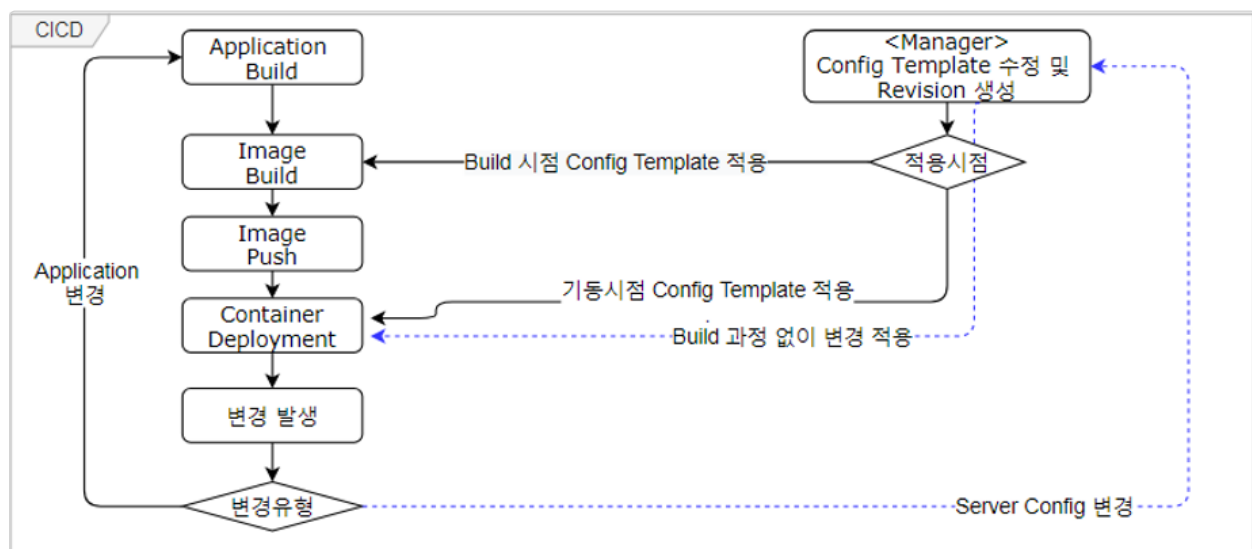


Figure 14. Service Cluster Configuration Management and Container Configuration Application

Server configuration must be pre-configured using LENA's Server Cluster function and reflected in Image or at startup time.

For detailed guidance on Server configuration, refer to the separately provided operator manual.

Container Image Build

Although LENA provides a Base Image, depending on project/customer policies or architecture standards, you may need to inherit from the Base Image or build a new one. In the architecture decision process, check the scope of changes to the LENA Image and establish image management policies/standards.

The following are reasons to build a new Base Image.

- OS does not match the LENA-provided default Image
- Change in LENA installation location (default: /usr/local/lena)
- Other mismatches between the target system's architecture standards and the LENA-provided default Image The gap cannot be resolved by configuration changes alone

In these cases, the Base Image must be regenerated through LENA technical support.

When only simple changes are needed as in the examples below, inherit the LENA Image and build the Base Image.

- Placement of Application Artifacts
- Installation of required Libraries
- JDK change (when the impact on the existing Image is no more than environment variable modifications)
- Modification of run command scripts
- Other mismatches between the target system's architecture standards and the LENA-provided default Image that can be resolved by configuration changes alone

For a detailed execution guide, refer to [Base Image Creation](#) in this document.

Application Deployment

From both Container and Server perspectives, consider the method of deploying Application Artifacts. Once the project's deployment procedure is determined, configure the Kubernetes Deployment Manifest or the ECS Task Definition according to the chosen approach.

From the WAS perspective, there are two methods to deploy an Application.

Server-side Deployment Method	Description
Deploy to default Deployment directory	Copy WAR or directory to \${LENA_SERVER_HOME}/webapps
Deploy according to individual Application settings	Configure individual Applications via Manager Copy WAR or directory to 'DocBase' location in the Application settings

From the Container perspective, there are two deployment methods.

Container Deployment Method	Description
Pre-bake into main Container Image	Copy/include Application Artifacts directly into the Base Image
Deploy at startup using Init Container	At startup, copy Artifacts embedded inside the Init Container or located in an external storage (Volume) into the main Container

2.7. Server Type Considerations – Embedded WAS

2.7.1. Deployment

Embedded WAS is deployed in Containers, and depending on configuration, single or multiple Instances are deployed simultaneously to Container platforms (Kubernetes, ECS, etc.).

1. Service Planning

Embedded WAS is deployed as single/multiple Containers, and to expose this to the outside or the Front-End, it is common to place a Service performing an L/B role in front. In Kubernetes, service types such as NodePort (serving specific Ports of installed Nodes), LoadBalancer (using external L/B), and Clusterip (internal fixed IP) are provided, while in ECS there is a method to specify ALB. Decide in advance how the Application will be serviced.

2. Number of Instances (Replica)

The number of Embedded WAS instances serving a single service varies with load, but the initial number of Instances to start must be defined in advance and reflected in the deployment configuration.

3. Service Mapping

In ECS, L/B can be directly specified in the Service, but in Kubernetes, mapping rules are defined based on key-value labels and mapped to Services. Establish a mapping standard that is non-duplicative across the entire system and convenient for operations, and reflect it in the deployment configuration.

2.7.2. Specifications

The specifications required to operate Embedded WAS are as follows.

Memory

Embedded WAS requires a minimum Heap Memory Size of 512Mbyte and the Image has the following default settings.

- Heap Memory : 1024 Mbyte
- Metaspace Memory : 128 Mbyte

To change these, adjust by setting the following environment variables.

- LENA_JVM_HEAP_SIZE
- LENA_JVM_METASPACE_SIZE



The values of the above environment variables are in MByte units and must be specified in the format of number+'m' (e.g., 1024m). If the format is inconsistent, it will not be applied.

Disk

Based on LENA Image, the Image size used is approximately 700 Mbyte, which is the total of OS + JDK + LENA + Library including the capacity of Image upper layers.

Additional Disk required for operation is calculated considering Log capacity when storing logs in file method and Application source file (Artifact) capacity.

2.7.3. Configuration

Network

1. Network Address

Embedded WAS has no special constraints on its network address. It should be placed to enable unidirectional communication towards Manager and Session Server on the network.

2. Service Port

Embedded WAS service ports are fixed as follows.

- HTTP service Port : 8180

These ports are fixed in the LENA Image. If you want to change the external service port, it is recommended to change it via port binding settings according to the basic philosophy of Containers rather than changing the LENA Image.

Environment Variables

The main environment variables applicable to Embedded WAS Containers are as follows.

Environment Variable	Description	Default Value	Changeable
LENA_SERVICE_PORT	• WAS service Port	8180	○
LENA_JVM_HEAP_SIZE	• Specify Heap Memory size	1024m	○
LENA_JVM_METASPACE_SIZE	• Metaspace Memory size	128m	○
LENA_JVM_OPTIONS	• User-defined JVM OPTION		○
LENA_MANAGER_ADDRESS	• Manager address • Format: IP / Domain address : service port		○
LENA_MANAGER_MONITORING_PORT	• Manager monitoring Port information	16100	○
LENA_CONFIG_TEMPLATE_ID	• Configuration file ID • Format: Service Cluster name		○
LENA_SPRING_PROFILES_ACTIVE	• SPRING PROFILE (Has the highest priority among PROFILE settings. This value must be set.)	default	○

Environment Variable	Description	Default Value	Changeable
LOG_OUTPUT_TYPE	<ul style="list-style-type: none"> LOG output type Allowed values: console or file 	console	<input type="radio"/>
LENA_LOG_OUTPUT_DIR	<ul style="list-style-type: none"> Log file creation location 	/usr/local/l ena/logs	<input type="radio"/>
LENA_SERVER_TYPE	<ul style="list-style-type: none"> Server type 	embedded	<input checked="" type="radio"/>
LENA_HOME	<ul style="list-style-type: none"> LENA installation Home Value: /usr/local/lena 	(See description)	<input checked="" type="radio"/>
LENA_SERVICE_ENDPOINT	<ul style="list-style-type: none"> Address of the service to which the WAS belongs 		<input type="radio"/>
LENA_APP_FILE	<ul style="list-style-type: none"> Application Jar file name 		<input type="radio"/>
LENA_APP_DIR	<ul style="list-style-type: none"> Application Jar directory name 	/usr/local/l ena	<input type="radio"/>
LENA_EXCEPTION_ALERT_ENABLE	<ul style="list-style-type: none"> Whether to collect information when an exception occurs 	false	<input type="radio"/>
LENA_EXCEPTION_CLASSES_PATTERNS	<ul style="list-style-type: none"> Target exception class patterns to collect; set multiple classes joined by ',' 		<input type="radio"/>
LENA_EXCEPTION_EXCLUDE_CLASS_PATTERNS	<ul style="list-style-type: none"> Exception class patterns to exclude; set multiple classes joined by ',' 		<input type="radio"/>
LENA_FULLSTACK_HOOKED_EXCEPTION_ENABLE	<ul style="list-style-type: none"> Whether to collect full stack trace when an exception occurs 	true	<input type="radio"/>
LENA_STUCKTHREAD_ALERT_ENABLE	<ul style="list-style-type: none"> Whether to collect information when a thread stuck occurs 	false	<input type="radio"/>
LENA_OOM_ALERT_ENABLE	<ul style="list-style-type: none"> Whether to collect information when Out Of Memory occurs 	true	<input type="radio"/>
LENA_FULLGC_ALERT_ENABLE	<ul style="list-style-type: none"> Whether to collect information when Full GC occurs 	false	<input type="radio"/>
LENA_REVERSE_TCP_CONNECTION_ENABLE	<ul style="list-style-type: none"> Whether to use Manager connection via Reverse TCP Connection 	true	<input type="radio"/>

Environment Variable	Description	Default Value	Changeable
LENA_CONFIG_SERVER_URI	<ul style="list-style-type: none"> Config Server URI to fetch properties files from a Git Repository using Spring Cloud Config. When this value is set, the URI and server.lena.config.enabled are set in bootstrap.properties to true. 		<input type="radio"/>

Directory Structure

Based on the LENA Image, the default installation location is '/usr/local/lena', and its sub-structure is as follows.

Directory (Under \${LENA_HOME})	Description	Notes
logs	Repository for LENA management log files	
etc/info	Repository for Image Build information files	

Log & Dump Output

Log supports both 'console' method that outputs to Standard Out / Error and 'file' method that outputs to files. You can switch the output method by setting the environment variable 'LOG_OUTPUT_TYPE' to 'console' or 'file'.

1. Console Output

This is commonly used in Container environments. The Server's Application Log, Access Log, and GC Log are all output to Standard Out. They are stored at designated locations on the Node (Host) by the log driver configured in Docker (default log file location in Docker: /var/lib/docker/containers/[container-id]/[container-id]-json.log), or can be collected/stored by a log aggregator such as FluentD for integrated management.

2. File Output

When file output is set, Log files and Dump files are saved as files under \${LENA_HOME}/servers/appServer/logs directory, and each log file is daily rolled by logrotate settings.

The types of output log files and the output file names are as follows.

Log Type	Output Location
Access Log	access_appServer_\${HOSTNAME}.log
GC Log	gc_appServer_\${HOSTNAME}.log
Application Log	appServer_lena-\${HOSTNAME}.out.log

These log files are rolled daily by logrotate installed on the guest OS.

When outputting file-based logs, it is common to configure a log management stack (ELK, EFK Stack, etc.) that collects/queries logs via a separate log aggregator. For this purpose, it is common to add a side-car container such as Fluent-Bit to collect logs.

Dump files are created at the following location.

- Dump Home : `${LENA_HOME}/servers/appServer/dumps/ ${HOSTNAME}`

The `${HOSTNAME}` directory is used to distinguish between containers when storing dump files in an external volume.

Storage locations by dump type are as follows.

- Heap Dump : `${DUMP_HOME}/hdump`
- Thread Dump : `${DUMP_HOME}/tdump`
- Service Dump : `${DUMP_HOME}/sdump`

Health Check

For basic Health Check content, refer to the [Manager Health Check](#) section in this document.

Generally, WAS uses Http Get method for Health Check, but LENA WAS's default health check method is set to a TCP port check method. This is because the basic LENA Image does not include a business application, and if a business application is installed, it is recommended to update the appropriate Http Get Health Check settings for that application. The default settings based on the provided Kubernetes Manifest files are as follows.

1. Readiness Check

- TCPSocketAction : port 8180
- initialDelaySeconds : xx (adjustment required according to application characteristics)
- timeoutSeconds : xx (adjustment required according to application characteristics)

2. Liveness Check

- TCPSocketAction : port 8180
- initialDelaySeconds : xx (adjustment required according to application characteristics)
- periodSeconds : xx (adjustment required according to application characteristics)



- For LENA WAS, the service port switches to the Listen state after normal server startup.
- Since a health check succeeds when the page used for the health check returns success, it is recommended to select and apply a page that can also determine whether the service's back-end (e.g., database) is normal.

Container Image Build

Although LENA provides a Base Image, depending on project/customer policies or architecture standards, you may need to inherit from the Base Image or build a new one. In the architecture decision process, check the scope of changes to the LENA Image and establish image management policies/standards.

The following are reasons to build a new Base Image.

- OS does not match the LENA-provided default Image
- Change in LENA installation location (default: `/usr/local/lena`)
- Other mismatches between the target system's architecture standards and the LENA-provided default Image The gap cannot be resolved by configuration changes alone

In these cases, the Base Image must be regenerated through LENA technical support.

When only simple changes are needed as in the examples below, inherit the LENA Image and build the Base Image.

- Placement of Application Artifacts
- Installation of required Libraries
- JDK change (when the impact on the existing Image is no more than environment variable modifications)
- Modification of run command scripts
- Other mismatches between the target system's architecture standards and the LENA-provided default Image that can be resolved by configuration changes alone

For a detailed execution guide, refer to [Base Image Creation](#) in this document.

Application Deployment

From both Container and Server perspectives, consider the method of deploying Application Artifacts. Once the project's deployment procedure is determined, configure the Kubernetes Deployment Manifest or the ECS Task Definition according to the chosen approach.

From the WAS perspective, there are two methods to deploy an Application.

Server-side Deployment Method	Description
Deploy to default Deployment directory	Set \${LENA_APP_FILE} Copy Jar into \${LENA_HOME}
Deploy according to individual Application settings	Set \${LENA_APP_FILE} and \${LENA_APP_DIR} Copy Jar into \${LENA_APP_DIR}

From the Container perspective, there are two deployment methods.

Container Deployment Method	Description
Pre-bake into main Container Image	Copy/include Application Artifacts directly into the Base Image
Deploy at startup using Init Container	At startup, copy Artifacts embedded inside the Init Container or located in an external storage (Volume) into the main Container

2.8. Server Type Considerations – Web Server

2.8.1. Deployment

Refer to "[WAS Deployment](#)" in this document.

2.8.2. Specifications

The specifications required to operate Web Server are as follows.

Memory

The Web Server requires at least 512Mbyte of heap memory, and the Agent requires 64~256MB of heap memory.

Disk

Based on the LENA Image, the image size used is about 900 Mbytes, which is the sum of OS + JDK + LENA + Library and includes the size of the upper layers of the image. Additional disk required for operations is calculated considering the log capacity when storing logs as files and the size of web resource files (artifacts).

2.8.3. Configuration

Network

1. Network Address

The Web Server has no special constraints on its network address. It must be configured to enable unidirectional communication to Manager and WAS or to the WAS service endpoint on the network.

2. Service Port

The Web Server service ports are fixed as follows.

- HTTP service Port : 7180
- HTTPS service Port : 7543

These ports are fixed in the LENA Image. If you want to change the external service, it is recommended to change it via port binding settings according to the basic philosophy of Containers rather than changing the LENA Image.

Environment Variables

The main environment variables applicable to the WEB Server Container are as follows.

Environment Variable	Description	Default Value	Changeable
LENA_MANAGER_ADDRESS	<ul style="list-style-type: none">• Manager address• Format: IP / Domain address : service port		○
LENA_MANAGER_KEY	<ul style="list-style-type: none">• Manager Open API access token		○
LENA_SERVICE_PORT	<ul style="list-style-type: none">• WAS service Port	7180	○
LENA_CONFIG_TEMPLATE_DOWNLOAD	<ul style="list-style-type: none">• Whether to download configuration files from Manager• Allowed values: Y or N		○

Environment Variable	Description	Default Value	Changeable
LENA_CONFIG_TEMPLATE_ID	<ul style="list-style-type: none"> Configuration file ID Format: Service Cluster name:Revision number 		<input type="radio"/>
LENA_LICENSE_DOWNLOAD_URL	<ul style="list-style-type: none"> License download URL Input value: manager or customer-owned License download URI 	manager	<input type="radio"/>
LENA_CONTRACT_CODE	<ul style="list-style-type: none"> Encrypted value for the contract code related to license issuance. 		<input type="radio"/>
LOG_OUTPUT_TYPE	<ul style="list-style-type: none"> LOG output type Allowed values: console or file 	console	<input type="radio"/>
LENA_LOG_OUTPUT_DIR	<ul style="list-style-type: none"> Log file creation location 	/usr/local/lenaw/servers/webServer/logs	<input type="radio"/>
LENA_AGENT_RUN	<ul style="list-style-type: none"> Whether to start Node Agent 	Y	<input type="radio"/>
LENA_SERVER_TYPE	<ul style="list-style-type: none"> Server type 	web	<input checked="" type="radio"/>
LENA_HOME	<ul style="list-style-type: none"> LENA installation Home Value: /usr/local/lena 	(See description)	<input checked="" type="radio"/>
LENA_SERVER_HOME	<ul style="list-style-type: none"> LENA server installation location <ul style="list-style-type: none"> Value: /usr/local/lenaw/servers/webServer 	(See description)	<input checked="" type="radio"/>
LENA_USER	<ul style="list-style-type: none"> OS user account to use for Manager startup 	root	<input type="radio"/>
LENA_USER_GROUP	<ul style="list-style-type: none"> OS user group to use for Manager startup 	root	<input type="radio"/>
LENA_HEALTH_CHECK	<ul style="list-style-type: none"> Whether to perform Health Check 	N	<input type="radio"/>
LENA_HEALTH_CHECK_FAILURE_THRESHOLD	<ul style="list-style-type: none"> Health Check failure threshold 	5	<input type="radio"/>
LENA_HEALTH_CHECK_TERM_EXECUTION	<ul style="list-style-type: none"> Whether to perform follow-up tasks when the Health Check failure threshold is exceeded 	true	<input type="radio"/>

Environment Variable	Description	Default Value	Changeable
LENA_HEALTH_CHECK_TERM_EXECUTION_SCRIPT	<ul style="list-style-type: none"> Follow-up task script information when the Health Check failure threshold is exceeded 	stop-container	<input type="radio"/>
LENA_HEALTH_CHECK_TERM_EXECUTION_INTERVAL	<ul style="list-style-type: none"> Follow-up task execution cycle when the Health Check failure threshold is exceeded 	300 (seconds)	<input type="radio"/>



- LENA_CONFIG_TEMPLATE_ID: The revision number can be omitted; when omitted, the Default Revision is downloaded.
- LENA_CONTRACT_CODE: Used for license validity verification; if invalid, the license download is canceled.
- LOG_OUTPUT_TYPE: When the server configuration file download is applied, the log settings of the downloaded configuration file are applied.
- LENA_AGENT_RUN: For the Web Server, the Agent must be running to register with the Manager and be monitored.

Directory Structure

Based on the LENA Image, the default installation location is '/usr/local/lenaw,' and its sub-structure is as follows.

Directory (Under \${LENA_HOME})	Description	Notes
bin	Node Agent's Start/Stop scripts	
depot	Local Repository for installation	Not used
etc	Other meta information and configuration files	
license	Directory for managing License information	
logs	Repository for LENA management log files	
modules <ul style="list-style-type: none"> └ lena-agent └ lena-web-pe 	Storage home for LENA-provided modules Path where modules required for Node Agent execution are located WEB Server Library	
servers/webServer <ul style="list-style-type: none"> └ bin └ cache └ conf └ hook └ htdocs └ logs └ temp 	Server installation Home, \${LENA_SERVER_HOME} Repository for server start/stop/management scripts Repository for cache information Repository for server configuration information Repository for life-cycle hook shell files Default web resource storage directory Repository for log files Working temporary directory	
tmp	LENA temporary management directory	

Log

Log supports both 'console' method outputting to Standard Out / Error and 'file' method outputting to files. You can switch the output method by setting the environment variable 'LOG_OUTPUT_TYPE' to 'console' or 'file'.

1. Console Output

Refer to [“Log & Dump Output”](#) in this document.

2. File Output

When file output is set, Log files and Dump files are stored as files under `${LENA_HOME}/servers/webServer/logs` directory, and each log file is daily rolled by logrotate settings.

The types of output log files and the output file names are as follows.

Log Type	Output File Name	Description
Error Log	error_webServer.log	Web server error log
Access Log	access_webServer.log	Access log
Trace Log	trace_webServer.log	Service tracing log, Not used for container-type servers
NTrace Log	ntrace_webServer.log	
LSC Log	lsc_webServer.log	

Health Check

Generally, WAS uses Http Get method for Health Check, but the default health check method of the LENA Web Server is set to TCP port check. This is because the basic LENA Image does not include a business application, and if a business application is installed, it is recommended to update the appropriate Http Get Health Check settings for that application. The default settings based on the provided Kubernetes Manifest files are as follows.

1. Readiness Check

- TCPSocketAction : port 7180
- initialDelaySeconds : xx (adjustment required according to application characteristics)
- timeoutSeconds : xx (adjustment required according to application characteristics)

2. Liveness Check

- TCPSocketAction : port 7180
- initialDelaySeconds : xx (adjustment required according to application characteristics)
- periodSeconds : xx (adjustment required according to application characteristics)

Server Configuration Management

Refer to [“Server Configuration Management”](#) in this document.

Container Image Build

Refer to [“Container Image Build”](#) in this document.

Chapter 3. Installation Common Requirements

3.1. Installation Preparation

- <Container Operating Environment Check>

For installation, an environment where Containers such as Kubernetes and ECS can operate must be configured first. Additionally, appropriate permissions and Profile settings (e.g., Kubernetes Config) to access the environment from CLI are required, and when installing Manager or Session Server on VM, the target VM for installation must be prepared.

- <Network Environment Check>

Communication between system configuration Servers must be verified. When all are installed within Container N/W, there are generally no communication restrictions, but when configured in VM mixed mode, prior confirmation of communication possibility between Manager Server, Web Server, WAS, and Session Server is required. The following are N/W Ports required for Service or mutual communication based on LENA Image.

Source	Target	Port	Purpose
WEB/WAS/Session Server	Manager	TCP 7700 TCP 16100 UDP 16100	Manager service provision Monitoring, registration
Bastion, Administrator PC	Manager	TCP 7700	Manager service provision
L/B , Front End	WEB Server	TCP 7180	Service provision
L/B , Front End	WAS	TCP 8180	Service provision
WAS	Session Server	TCP 5180	Service provision

- <Image Access Environment Check>

To utilize LENA public Images, the Container platform must have access to Docker Hub. If access is not possible, you must Pull the LENA Image in an accessible environment (docker pull), save it as a file (docker save), and load it (docker load) in an environment accessible to that platform. Below is an example of handling this process.

```
$ sudo docker pull lenacloud/lena-cluster:{TAG_NAME}
... <output omitted>
$ sudo docker save -o lena-cluster.tar lenacloud/lena-cluster:{TAG_NAME}
... <output omitted>
$ sudo docker load -i lena-cluster.tar
... <output omitted>
```

3.2. Base Image Creation

For WAS or WEB Server cases, there may be a need to regenerate the Base Image. The following explains creating (building) a Base Image by inheriting LENA Image. Base Image creation is the entire

process of writing a Dockerfile that utilizes LENA Image, creating a Docker Image from this file, and registering it in a shared Repository.

The expected major changes to the Base Image include the following cases:

1. Adding Required Libraries
Install using OS-specific installation commands (yum, apt-get, etc.) or install directly.
2. Changing JDK
When wanting to change the JDK of LENA Base Image, install using installation commands or direct extraction method.
3. Application Deployment
The default location for Application Deployment is '/usr/local/lena/server/appServer/webapps'.
You can deploy by directly copying WAR files or Exploded Directories here. In this case, the WAR file name or Directory name is designated as the Context Path.
Alternatively, you can deploy by copying WAR or Directory to a specified location (DocBase) through individual Application configuration. In this case, the Application is served with the Context Path specified in the configuration.
For detailed configuration, refer to the Application configuration section of the separately provided 'Operator Manual'.
4. Modifying Execution Command Script
LENA's default Container execution Command is `${LENA_HOME}/docker-entrypoint.sh`. This is a Shell Script that executes environment configuration based on initial environment variables, Server configuration file download, License download, Server execution, etc. When changes are needed to fit the Project environment, modify and apply this script.

3.2.1. Base Image Creation Procedure

The Base Image creation procedure is as follows:

1. Dockerfile creation
2. Docker Image build
3. Docker Image registration

Dockerfile Creation

The following code is a Dockerfile example for creating a Base Image, written based on a CentOS-based Application Server.

Dockerfile Sample

```

# This is a sample of Dockerfile to build project's Base Image with LENA
template.
# LENA image provides just basic environment to run LENA WAS.
# The project and the customer are responsible for optimization or change the
environments.
# Before building it, you need to check the proper LENA image repository &
tag.
FROM docker.io/lenacloud/lena-cluster:{TAG_NAME}

# Change or add JDK & packages as your own policy.
# RUN yum update -y
# RUN yum install -y java-1.8.0-openjdk-devel.x86_64

# The service address of LENA manager.
ENV LENA_MANAGER_ADDRESS lena-manager.namespace.svc.cluster.local:7700

# The key to access LENA manager.
ENV LENA_MANAGER_KEY you_manager_key_from_manager_user_admin_menu

# Id of template. Format is <Service Container Name>:<Version>.
ENV LENA_CONFIG_TEMPLATE_ID was-cluster_01:1

# To download and validate your license, LENA_CONTRACT_CODE is required.
# You can acquired it via LENA supplier.
# ENV LENA_CONTRACT_CODE your_own_lena_contract_code

# Download & change template files from manager.
RUN ${LENA_HOME}/docker-entrypoint.sh download_template

# If you have your own license file, copy it.
#COPY license.xml ${LENA_HOME}/license/

# Or If you uploaded your own container license file to manager, you can
download it from manager
# Caution!. After downloading you need to validate the file with like
'xmllint' command.
# RUN ${LENA_HOME}/docker-entrypoint.sh download_license

# Copy your application source to deployment path.
# COPY application.war /usr/local/lena/servers/appServer/webapps/

```

The structure of the above sample Dockerfile is as follows:

Table 1. Item-by-item explanation of sample Dockerfile

Configuration Order	Description
Specifying Parent Layer Image	<p><Syntax> FROM \${Parent Image}</p> <p>Parent Image should be LENA Image or an Image that inherits from LENA Image</p> <p>※ Enter the Repository/Tag of the Base Image selected by the Project.</p>
Additional Library Installation	<p><Syntax> RUN yum install \${Additional Library}</p> <p>Add necessary Libraries using package installation commands by OS, and if separate installation methods are needed for specific packages, install additional Libraries according to the package's guide.</p> <p><JDK Reinstallation></p> <p>JDK can also be installed in the same way as adding Libraries. However, since LENA's Servers reference JDK, the following configuration information must also be changed:</p> <p>* The Symbolic Link '/usr/lib/jvm/java' must be reconfigured to match the installed JDK location.</p> <p>This information is used in the following parts:</p> <ul style="list-style-type: none"> • Environment variable \${JAVA_HOME} • JDK installation location: \${LENA_HOME}/etc/info/ java-home.info value • Server environment configuration: JAVA_HOME value in \${LENA_HOME}/env.sh • Domain cache value TTL setting in docker-entrpoint.sh
Environment Variable Configuration	<p><Syntax> ENV \${key} \${value}</p> <p>Set environment variables to download configuration information or license below. Values set here can also be used at Container startup time.</p> <ul style="list-style-type: none"> • LENA_MANAGER_ADDRESS : Service address of installed Manager • LENA_MANAGER_KEY : Authentication token required for Manager access via Open API (Can be checked in Manager's Admin > Users menu) • LENA_CONFIG_TEMPLATE_ID : Configuration information identifier to download, (Service Cluster name + ":" + Revision number) • ENV LENA_CONTRACT_CODE : Contract code for license download
Configuration Information Download	<p><Syntax> RUN \${LENA_HOME}/docker-entrpoint.sh download_template</p> <p>Download configuration information from Manager. Call Open API with commands like curl / wget.</p> <p>After Manager download, execute archive extraction, Mod/Owner/Group modification.</p>

Configuration Order	Description
License Copy or Download	<p><Syntax> RUN curl -o W\${LENA_HOME}/license/license.xml ...<omitted below></p> <p>Download License from Manager. After download, perform XML Validation and Owner/Group modification.</p> <p>COPY license.xml \${LENA_HOME}/license/</p> <p>If there is a separately issued License, copy it inside the Container.</p>
Application Artifact Copy	<p><Syntax> COPY application.war \${LENA_SERVER_HOME}/webapps/</p> <p>Copy Application Artifact (application.war in the example) to the Deploy location. The default locations are as follows:</p> <ul style="list-style-type: none"> • WAS : \${LENA_SERVER_HOME}/webapps/ • WEB Server : \${LENA_SERVER_HOME}/webapps/htdocs <p>These locations can be changed by individual Server and Application configuration.</p>

Based on the completed Dockerfile, you can Build an Image (docker build), and after building, register it (docker register) in a Registry such as Docker Hub or ECR to make it available for use in Container Platform.

Dockerfile Creation (Regular Account)

When you want to use a regular account instead of root account, additional work is required when creating Base Image. The sample below uses CentOS7 and the 'lena' regular account case. It proceeds in the order of account creation / sudo installation. To enable root privilege commands in the entrypoint script during Container startup, this process grants sudo privileges and configures sudo commands to work without password settings. When the entrypoint script ends, it changes to require password input when using sudo, restricting sudo commands for regular accounts.

Dockerfile Sample

```
FROM docker.io/lenacloud/lena-cluster:{TAG_NAME}

# create account
ENV LENA_USER lena
RUN adduser ${LENA_USER}
RUN chown -R ${LENA_USER}:${LENA_USER} ${LENA_HOME}

# sudo auth setting
RUN yum install -y sudo && yum clean all
RUN usermod -aG wheel ${LENA_USER}
## nopasswd sudo on - off this option in docker entry point
RUN sed -i 's/# %wheel/%wheel/g' /etc/sudoers

USER ${LENA_USER}
```

Docker Image Build

The following is the command to Build an Image from a Dockerfile. You must define the Image's Repository and Tag Rule in advance, and note that the last argument specifies the location of the Dockerfile. (The example below is valid when the Dockerfile is in the Current Path.)

Docker Image build execution command (example)

```
$ docker build --no-cache --rm --tag [REPOSITORY[:TAG]] .
```

The options used in the above command are as follows. Select and use options according to the Project environment.

- `--no-cache` : Do not use cache generated from previous builds
- `--rm` : Delete temporary containers when image creation succeeds

Docker Image Registration

The following is the command to register an Image in the Image Registry. It requires environment configuration such as users and passwords with permissions to register in the Registry in advance.

Docker Image registration execution command (example)

```
$ docker push [OPTIONS] [REPOSITORY[:TAG]]
```

For other detailed Docker management commands, refer to the following official site:

<https://docs.docker.com/engine/reference/commandline/docker/>

`docker commit` is a command that registers a currently running Docker Container as an Image.

Register running Container as Image (example)

```
$ docker commit [OPTIONS] CONTAINER [REPOSITORY[:TAG]]
```


Chapter 4. Kubernetes-based Deployment

4.1. Common Deployment Considerations

4.1.1. Deployment Preparation

A Kubernetes runtime environment must be configured prior to LENA installation. The following items must be prepared:

1. Configuration of Kubernetes Cluster and Namespace to run LENA
2. Environment with kubectl installed and CLI capability
3. Kubernetes configuration (~/.kube/config) to enable access to the Kubernetes Cluster
4. Accessible Docker Repository (e.g., ECR, Docker Hub, internal Docker Repository)
5. (Optional) Kubernetes Config file: Required to be uploaded to Manager for Log and Terminal access in LENA Manager
6. LENA Container License and contract code

4.1.2. Deployment Execution

While Kubernetes offers various deployment methods for different orchestration tools, this document explains the process based on using kubectl, which is the basic tool.

Kubernetes performs deployment/deletion/updates using YAML-format manifest files that describe resource deployment methods, environment configuration information, and specifications. The `{manifest-file.yaml}` described below refers to this manifest. This document describes only the minimum commands and options required for deployment, so refer to the published Kubernetes documentation for detailed information.

Setting the Working Namespace

Since all deployment-related tasks below are performed on a namespace basis, you must set the target namespace before performing deployment.

```
$ kubectl config set-context --current --namespace={namespace}
```

If not configured this way, you must add the `'--namespace={namespace}'` clause after all execution commands below. Here is an example:

```
$ kubectl apply -f {manifest-file.yaml} --namespace={namespace}
```

Kubernetes Resource Deployment and Updates

Execute deployment using the `'kubectl apply'` command, with the following format:

```
$ kubectl apply -f {manifest-file.yaml}
```

This command not only deploys but also updates the specifications of the resource if an object with the same name exists in the same namespace.

Deleting Kubernetes Resources

Delete deployed resources using the 'kubectl delete' command.

```
$ kubectl delete -f ${manifest-file.yaml}
```

Workload Updates and Rollbacks

For Applications and Web Servers, workload updates may be necessary due to changes in Application Artifacts. Kubernetes provides a method to update these using the Rolling update approach.

The following is the command to update deployed workloads using the Rolling method:

```
$ kubectl rollout restart ${workloadType}/${workloadName}
```

This command refreshes the workload and creates sequential revisions. The command to view the revision list is as follows:

```
$ kubectl rollout history ${workloadType}/${workloadName}
```

The following is the command to rollback to the previous state:

```
$ kubectl rollout undo ${workloadType}/${workloadName}
```

The following is the command to rollback to a specified revision:

```
$ kubectl rollout undo ${workloadType}/${workloadName}  
--to-revision=${revisionNo}
```

In the above command, the argument `${workloadType}` is the workload type, with values being one of 'statefulset', 'deployment', or 'daemonset'. `${workloadName}` is the name of the workload, and `${revisionNo}` is the revision value.

Verifying Deployed Resources

The command to check deployed Kubernetes objects such as workloads and services is `kubectl get`.

```
$ kubectl get ${resourceType}
```

In the above command, `${resourceType}` is one of the Kubernetes resource types such as 'statefulsets', 'deployments', 'daemonsets', 'services', 'configMaps'.

4.2. Manager Deployment

4.2.1. Configuration Items

Basic Configuration Items

Manager Container should be deployed based on the following recommended settings:

Configuration Item	Configuration Value / Description	Notes
Workload Type	StatefulSet	-
Replica Count	1	-
Container Port	TCP : 7700 UDP : 16100 TCP : 16100	-
Volume Mount	Map directory /usr/local/lena/repository to external Volume (mapped using persistentVolumeClaim method in the example below)	-
Probe (Health Check)	HttpGetAction, call '/lena' page	-

Application-Time Decision Items – Workload Related

Configuration elements that must be determined and applied according to the project environment at application time, and sample values applied to the Manifest file described later are as follows:

Configuration Item	Description	Sample Value
Container Image	LENA Manager Image matching OS and JDK version selected according to project-specific architecture decisions	lenacloud/lena-manager.{TAG_NAME}
namespace	Logical group name within Kubernetes	default
name	Name of the workload, this value is used as the prefix for Pod name / Hostname	lena-manager
label	Label used for connection with Service and search, defined as Key: Value pairs	type: sample-lena-manager
imagePullPolicy	Image pull policy, select one of the following: <ul style="list-style-type: none"> • Always : Always pull from Repository • IfNotPresent : Pull only when Image is not available locally • Never : Do not pull 	Always

Configuration Item	Description	Sample Value
Other Environment Variables	Environment variables can be set via ENV or Config Map	(configMap method)

The configurable environment variables are as follows. For detailed descriptions of each environment variable, refer to the [Manager Environment Variables](#) section of this document.

Environment Variable	Description	Sample Value
LENA_JVM_HEAP_SIZE	Specify Heap Memory size	1024m (default)
LENA_JVM_METASPACE_SIZE	Metaspace Memory size	256m (default)
LENA_MANAGER_DOMAIN_ENABLED	Domain Name activation status	Y
LENA_MANAGER_ADDRESS	Service Domain address : port	lena-manager.default.svc.cluster.local:7700
JAVA_DOMAIN_CACHE_TTL	Domain address cache time (seconds)	3 (default)

Application-Time Decision Items – Service Related

Configuration Item	Description	Sample Value
namespace	Logical group name within Kubernetes	default
name	Name identifying the Service, must be unique within the namespace. This value is applied to the Service Domain address.	lena-manager
type	Method for exposing Service externally, one of the following: <ul style="list-style-type: none"> NodePort : Service through specified port on all Nodes where k8s is installed LoadBalancer : Service through external Loadbalancer ClusterIp : Dedicated for k8s Cluster, service with fixed IP 	NodePort

4.2.2. Manifest-based Deployment

Workload

Containers in Kubernetes are installed in Pod units, and the typical method is to install by creating a Manifest file in YAML file format that describes Kubernetes Objects.

The following is a sample Manifest file for installing Manager, and the detailed content can be modified to suit the project environment according to the configuration item decisions described earlier.

LENA Manager Workload Specification (Manifest) File

```
---
---
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: lena-manager
spec:
  selector:
    matchLabels:
      type: lena-manager
  serviceName: lena-manager
  replicas: 1
  template:
    metadata:
      labels:
        type: lena-manager
    spec:
      containers:
        - name: lena-manager
          image: docker.io/lenacloud/lena-manager:{TAG_NAME}
          imagePullPolicy: Always
          ports:
            - containerPort: 7700
          envFrom:
            - configMapRef:
                name: configmap-lena-manager
          volumeMounts:
            - name: wsy-lena-manager-repository
              mountPath: /usr/local/lena/repository
          readinessProbe:
            httpGet:
              path: /lena
              port: 7700
            initialDelaySeconds: 20
            timeoutSeconds: 1
          livenessProbe:
            httpGet:
              path: /lena
              port: 7700
            initialDelaySeconds: 30
            periodSeconds: 5
      volumes:
        - name: wsy-lena-manager-repository
```

```
    persistentVolumeClaim:
      claimName: lena-manager-repository
      terminationGracePeriodSeconds: 0

---
apiVersion: v1
kind: ConfigMap
metadata:
  name: configmap-lena-manager
data:
  LENA_MANAGER_DOMAIN_ENABLED: "Y"
  LENA_MANAGER_ADDRESS: "lena-manager.default.svc.cluster.local:7700"
```

- Deployment Execution
Deployment is executed with the `kubectl apply` command. If the filename is `lena-manager-deployment-sample.yaml`, the deployment command is as follows:

```
$ kubectl apply -f lena-manager-deployment-sample.yaml
```

- Deployment Result Verification
The deployed workload can be verified by executing the `kubectl get` command.

```
$ kubectl get statefulsets
```

NAME	READY	AGE
lena-manager	1/1	10s

Service

The following is the Manifest file for deploying Manager Service:

LENA Manager Service Specification (Manifest) File

```

---
apiVersion: v1
kind: Service
metadata:
  name: lena-manager
spec:
  selector:
    type: lena-manager
  ports:
    - name: manager-tcp
      port: 7700
      targetPort: 7700
      nodePort: 31848
      protocol: TCP
    - name: monitoring-tcp
      port: 16100
      targetPort: 16100
      protocol: TCP
    - name: monitoring-udp
      port: 16100
      targetPort: 16100
      protocol: UDP
  type: NodePort

```

- Deployment Execution

Deployment is executed with the `kubectl apply` command. If the filename is `lena-manager-service-sample.yaml`, the deployment command is as follows:

```
$ kubectl apply -f lena-manager-service-sample.yaml
```

- Deployment Result Verification

The deployed workload can be verified by executing the `kubectl get` command.

```

$ kubectl get services
NNAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
...
lena-manager NodePort 10.43.xx.xx <none> 7700:30770/TCP,16100:31610/UDP
10s

```

4.2.3. Manager Access

If the Service type is set to NodePort, access the Manager by connecting to `http://[Node IP]:[Node Port] /`.

4.3. Session Server Deployment

4.3.1. Deployment Preparation

To register Session Server with Manager before deployment, you must register a 'Service Cluster' in Manager. Create a new Service Cluster of Session Server type in the Manager's 'Cluster > Session Cluster' menu location.

For detailed information on Service Cluster creation and management, refer to the separately provided 'Administrator Manual' document.

4.3.2. Configuration Items

Basic Configuration Items

Session Server Container should be deployed based on the following recommended settings:

Configuration Item	Configuration Value / Description	Notes
Workload Type	StatefulSet	-
Replica Count	2 (Primary, Secondary Server 2 units)	-
Container Port	TCP : 5180	-
Probe	ExecAction, call \${LENA_SERVER_HOME}/health.sh	-

Application-Time Decision Items – Workload Related

Configuration elements that must be determined and applied according to the project environment at application time, and sample values applied to the Manifest file described later are as follows:

Configuration Item	Description	Sample Value
Container Image	LENA Manager Image matching OS and JDK version selected according to project-specific architecture decisions	lenacloud/lena-session:{TAG_NAME}
namespace	Logical group name within Kubernetes	default
name	Name of the workload, this value is used as the prefix for Pod name / Hostname	lena-session
label	Label used for connection with Service and search, defined as Key: Value pairs	type: lena-session
imagePullPolicy	Image pull policy	Always
Other Environment Variables	Environment variables can be set via ENV or Config Map	(configMap method)

The configurable environment variables are as follows. For detailed descriptions of each environment

variable, refer to the [Session Environment Variables](#) section of this document.

Environment Variable	Description	Sample Value
LENA_JVM_HEAP_SIZE	• Specify Heap Memory size	1024m (default)
LENA_CONFIG_TEMPLATE_ID	• Service Cluster name : Revision No	SESSION-001
LENA_MANAGER_ADDRESS	• Service Domain address : port	sample-lena-manager.default.svc.cluster.local:7700
JAVA_DOMAIN_CACHE_TTL	• Domain address cache time (seconds)	0 (default)
LENA_SESSION_0_ADDRESS	• Primary Session server address, must match StatefulSet configuration	lena-session-0.default.svc.cluster.local
LENA_SESSION_1_ADDRESS	• Secondary Session server address, must match StatefulSet configuration	lena-session-1.default.svc.cluster.local
LENA_SESSION_EXPIRE_SEC	• Session expiration time (seconds)	1800 (default)
LENA_CONFIG_SHARE_SESSION	• Session sharing between Applications	N

Application-Time Decision Items – Service Related

Configuration Item	Description	Sample Value
namespace	Logical group name within Kubernetes	default
name	Name identifying the Service, must be unique within the namespace. This value is applied to the Service Domain address.	lena-session
type	Method for exposing Service externally, not specified for Session Server.	

4.3.3. Manifest-based Deployment

Workload

Containers in Kubernetes are installed in Pod units, and the typical method is to install by creating a Manifest file in YAML file format that describes Kubernetes Objects.

The following is a sample Manifest file for installing Session Server, and the detailed content can be modified to suit the project environment according to the configuration item decisions described earlier.

LENA Session Workload Specification (Manifest) File Example

```
---
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: lena-session
spec:
  selector:
    matchLabels:
      type: lena-session
  serviceName: lena-session
  replicas: 2
  template:
    metadata:
      labels:
        type: lena-session
    spec:
      containers:
        - name: lena-session
          image: docker.io/lenacloud/lena-session:{TAG_NAME}
          imagePullPolicy: Always
          ports:
            - containerPort: 5180
          envFrom:
            - configMapRef:
                name: configmap-lena-session
          tty: true
          readinessProbe:
            exec:
              command:
                - /usr/local/lena/servers/sessionServer/health.sh
            initialDelaySeconds: 20
            periodSeconds: 5
          livenessProbe:
            exec:
              command:
                - /usr/local/lena/servers/sessionServer/health.sh
            initialDelaySeconds: 30
            periodSeconds: 5
      terminationGracePeriodSeconds: 0
---
apiVersion: v1
kind: ConfigMap
metadata:
  name: configmap-lena-session
data:
  LENA_SESSION_0_ADDRESS: "lena-session-0.lena-
```

```
session.default.svc.cluster.local:5180"
LENA_SESSION_1_ADDRESS: "lena-session-1.lena-
session.default.svc.cluster.local:5180"
LENA_MANAGER_ADDRESS: "lena-manager.default.svc.cluster.local:7700"
LENA_SESSION_EXPIRE_SEC: "1800"
LENA_CONFIG_TEMPLATE_ID: "SESSION-001"
LENA_CONFIG_SHARE_SESSION: "N"
```

- Deployment Execution

Deployment is executed with the `kubectl apply` command. If the filename is `lena-session-deployment-sample.yaml`, the deployment command is as follows:

```
$ kubectl apply -f lena-session-deployment-sample.yaml
```

- Deployment Result Verification

The deployed workload can be verified by executing the `kubectl get` command.

```
$ kubectl get statefulsets
NAME READY AGE
lena-manager 1/1 30m
lena-session 2/2 10s
```

Service

The following is the Manifest file for deploying Session Service:

LENA Session Service Specification (Manifest) File Example

```
---
apiVersion: v1
kind: Service
metadata:
  name: lena-session
spec:
  selector:
    type: lena-session
  clusterIP: None
```

- Deployment Execution

Deployment is executed with the `kubectl apply` command. If the filename is `lena-session-service-sample.yaml`, the deployment command is as follows:

```
$ kubectl apply -f lena-session-service-sample.yaml
```

- Deployment Result Verification

The deployed workload can be verified by executing the `kubectl get` command.

```
$ kubectl get services
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
...
lena-session ClusterIP None <none> <none> 10s
...
```

4.3.4. Server Registration Verification

Access Manager and select the corresponding Service Cluster from the 'Cluster > Service Cluster' menu to verify proper registration. Click the **ServerList** button to verify that 2 Session Servers are displayed at the bottom.



Servers are registered by Manager's Scheduler, so they are automatically registered within a maximum of 15 seconds.

4.4. WAS Deployment

4.4.1. Deployment Preparation

To register WAS Server with Manager before deployment, you must register a 'Service Cluster' in Manager. Create a new Service Cluster of WAS(Enterprise/SE) type in the Manager's 'Cluster > Server Cluster' menu location.

After creating the Service Cluster, set the Service Endpoint in the Overview tab. For Kubernetes, the format becomes "http://\$W{Service name}.\$W{namespace name}.svc.cluster.localhost:\${Service port}". This value is utilized in the Proxy settings of the Web Server's VirtualHost.

Configure the WAS settings in the Template tab of the created Service Cluster. After saving the configuration information, create a Revision for the Template. For detailed information on Service Cluster creation and management, refer to the separately provided 'Administrator Manual' document.

4.4.2. Configuration Items

Basic Configuration Items

WAS Container should be deployed based on the following recommended settings:

Configuration Item	Configuration Value / Description	Notes
Workload Type	Deployment	
Container Port	TCP : 8180	

Application-Time Decision Items – Workload Related

Configuration elements that must be determined and applied according to the project environment at application time, and sample values applied to the Manifest file described later are as follows:

Configuration Item	Description	Sample Value
Container Image	LENA Manager Image matching OS and JDK version selected according to project-specific architecture decisions	lenacloud/lena-cluster:{TAG_NAME}
namespace	Logical group name within Kubernetes	default
name	Name of the workload, this value is used as the prefix for Pod name / Hostname	lena-was
label	Label used for connection with Service and search, defined as Key: Value pairs	type: lena-was
strategy:type	Deployment Update policy, specify the Update policy name selected from RollingUpdate or Recreate	RollingUpdate
imagePullPolicy	Image pull policy	Always
replica count	Container (Pod) count	2
Probe	HttpGetAction, check page, start delay time, and period need to be set according to Application characteristics	'/' call
Other Environment Variables	Environment variables can be set via ENV or Config Map	(configMap method)

The configurable environment variables are as follows. For detailed descriptions of each environment variable, refer to the [WAS Environment Variables](#) section of this document.

Environment Variable	Description	Sample Value
LENA_CONFIG_TEMPLATE_ID	• Service Cluster name : Revision No	WAS-001:1
LENA_MANAGER_ADDRESS	• Service Domain address : port (Service address of previously installed Manager)	lena-manager.default.svc.cluster.local:7700
LENA_MANAGER_KEY	• LENA_MANAGER_KEY : Authentication token required for Manager access via Open API • Can be checked in Manager's Admin > Users menu	(Individual verification required in Manager)
LENA_CONFIG_TEMPLATE_DOWNLOAD	• Configuration information download status	Y
LENA_CONTRACT_CODE	• Contract code for license download • Code value provided when license is issued	(Individual code verification required)

Environment Variable	Description	Sample Value
LENA_LICENSE_DOWNLOAD_URL		manager
JAVA_DOMAIN_CACHE_TTL	• Domain address cache time (seconds)	3 (default)

Application-Time Decision Items – Service Related

Configuration Item	Description	Sample Value
namespace	Logical group name within Kubernetes	default
name	Name identifying the Service, must be unique within the namespace. This value is applied to the Service Domain address.	lena-was
type	Method for exposing Service externally, no separate configuration needed if there is no requirement to expose externally	

4.4.3. Manifest-based Deployment

Workload

Containers in Kubernetes are installed in Pod units, and the typical method is to install by creating a Manifest file in YAML file format that describes Kubernetes Objects.

The following is a sample Manifest file for installing WAS, and the detailed content can be modified to suit the project environment according to the configuration item decisions described earlier.

LENA WAS Workload Specification (Manifest) File Example

```
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: lena-was
spec:
  selector:
    matchLabels:
      type: lena-was
  replicas: 2
  strategy:
    type: RollingUpdate
  minReadySeconds: 10
  revisionHistoryLimit: 1
  template:
    metadata:
      labels:
        type: lena-was
```

```

spec:
  containers:
  - name: lena-was
    image: docker.io/lenacloud/lena-cluster:{TAG_NAME}
    imagePullPolicy: Always
    ports:
    - containerPort: 8180
    envFrom:
    - configMapRef:
        name: configmap-lena-was
    readinessProbe:
      httpGet:
        path: /
        port: 8180
        initialDelaySeconds: 10
        periodSeconds: 5
    livenessProbe:
      httpGet:
        path: /
        port: 8180
        initialDelaySeconds: 15
        periodSeconds: 5
    volumes:
  terminationGracePeriodSeconds: 0

---
apiVersion: v1
kind: ConfigMap
metadata:
  name: configmap-lena-was
data:
  LENA_CONFIG_TEMPLATE_DOWNLOAD: "Y"
  LENA_CONFIG_TEMPLATE_ID: "WAS-001"
  LENA_MANAGER_ADDRESS: "lena-manager.default.svc.cluster.local:7700"
  LENA_MANAGER_KEY:
"aSw7RMPSw15LeN%2FMZnrxzjgV0BzZe18iVHZbJ8CkdL1ea2Ecd8AleK9oPCLXuw%3D%3D"
  LENA_LICENSE_DOWNLOAD_URL: "manager"
  LENA_CONTRACT_CODE: "pghzJJqTdzaGtTuASr8yfw=="
  JAVA_DOMAIN_CACHE_TTL: "3"

```

- Deployment Execution
Deployment is executed with the `kubectl apply` command. If the filename is `lena-was-deployment-sample.yaml`, the deployment command is as follows:

```
$ kubectl apply -f lena-was-deployment-sample.yaml
```

- Deployment Result Verification
The deployed workload can be verified by executing the `kubectl get` command.

```
$ kubectl get deployments
NAME READY AGE
lena-was 2/2 10s
```

Service

The following is the Manifest file for deploying Application Service:

LENA WAS Service Specification (Manifest) File Example

```
---
apiVersion: v1
kind: Service
metadata:
  name: lena-was
spec:
  selector:
    type: lena-was
ports:
- port: 8180
  targetPort: 8180
```

- **Deployment Execution**
Deployment is executed with the kubectl apply command. If the filename is lena-was-service-sample.yaml, the deployment command is as follows:

```
$ kubectl apply -f lena-was-service-sample.yaml
```

- **Deployment Result Verification**
The deployed workload can be verified by executing the kubectl get command.

```
$ kubectl get services
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
...
lena-was ClusterIP 10.43.xx.xx <none> 8180/TCP 20s
```

4.4.4. Server Registration Verification

Access Manager and select the corresponding Service Cluster from the 'Cluster > Service Cluster' menu to verify proper registration. Click the **Server List button** to verify that 2 WAS are displayed at the bottom.



Servers are registered by Manager's Scheduler, so they are automatically registered within a maximum of 15 seconds.

4.5. Embedded WAS Deployment

4.5.1. Deployment Preparation

To register Embedded WAS Server with Manager before deployment, you must register a 'Service Cluster' in Manager. Create a new Service Cluster of WAS(Embedded) type in the Manager's 'Cluster > Server Cluster' menu location.

4.5.2. Configuration Items

Basic Configuration Items

WAS Container should be deployed based on the following recommended settings:

Configuration Item	Configuration Value / Description	Notes
Workload Type	Deployment	
Container Port	TCP : 8180	

Application-Time Decision Items – Workload Related

Configuration elements that must be determined and applied according to the project environment at application time, and sample values applied to the Manifest file described later are as follows:

Configuration Item	Description	Sample Value
Container Image	LENA Manager Image matching OS and JDK version selected according to project-specific architecture decisions	lenacloud/lena-embedded:{TAG_NAME}
namespace	Logical group name within Kubernetes	default
name	Name of the workload, this value is used as the prefix for Pod name / Hostname	lena-was
label	Label used for connection with Service and search, defined as Key: Value pairs	type: lena-was
strategy:type	Deployment Update policy, specify the Update policy name selected from RollingUpdate or Recreate	RollingUpdate
imagePullPolicy	Image pull policy	Always
replica count	Container (Pod) count	2
Probe	HttpGetAction, check page, start delay time, and period need to be set according to Application characteristics	'/' call

Configuration Item	Description	Sample Value
Other Environment Variables	Environment variables can be set via ENV or Config Map	(configMap method)

The configurable environment variables are as follows. For detailed descriptions of each environment variable, refer to the [Embedded WAS Environment Variables](#) section of this document.

Environment Variable	Description	Sample Value
LENA_CONFIG_TEMPLATE_ID	• Service Cluster name : Revision No	WAS-001:1
LENA_MANAGER_ADDRESS	• Service Domain address : port (Service address of previously installed Manager)	lena-manager.default.svc.cluster.local:7700
LENA_MANAGER_MONITORING_PORT	• Manager monitoring Port information	16100
LENA_APP_FILE	• Application Jar file name	sample-app.jar
LENA_APP_DIR	• Application Jar directory name	/usr/local/lena

Application-Time Decision Items – Service Related

Configuration Item	Description	Sample Value
namespace	Logical group name within Kubernetes	default
name	Name identifying the Service, must be unique within the namespace. This value is applied to the Service Domain address.	lena-was
type	Method for exposing Service externally, no separate configuration needed if there is no requirement to expose externally	

4.5.3. Manifest-based Deployment

Workload

Containers in Kubernetes are installed in Pod units, and the typical method is to install by creating a Manifest file in YAML file format that describes Kubernetes Objects.

The following is a sample Manifest file for installing Embedded WAS, and the detailed content can be modified to suit the project environment according to the configuration item decisions described earlier.

LENA Embedded WAS Workload Specification (Manifest) File Example

```
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: lena-was
spec:
  selector:
    matchLabels:
      type: lena-was
  replicas: 2
  strategy:
    type: RollingUpdate
  minReadySeconds: 10
  revisionHistoryLimit: 1
  template:
    metadata:
      labels:
        type: lena-was
    spec:
      containers:
        - name: lena-was
          image: docker.io/lenacloud/lena-embedded:{TAG_NAME}
          imagePullPolicy: Always
          ports:
            - containerPort: 8180
          envFrom:
            - configMapRef:
                name: configmap-lena-was
          readinessProbe:
            httpGet:
              path: /
              port: 8180
            initialDelaySeconds: 10
            periodSeconds: 5
          livenessProbe:
            httpGet:
              path: /
              port: 8180
            initialDelaySeconds: 15
            periodSeconds: 5
      volumes:
      terminationGracePeriodSeconds: 0

---
apiVersion: v1
kind: ConfigMap
metadata:
  name: configmap-lena-was
```

data:

LENA_CONFIG_TEMPLATE_ID: "WAS-001"

LENA_MANAGER_ADDRESS: "lena-manager.default.svc.cluster.local:7700"

LENA_APP_FILE: "sample-app.jar"

- Deployment Execution

Deployment is executed with the `kubectl apply` command. If the filename is `lena-was-deployment-sample.yaml`, the deployment command is as follows:

```
$ kubectl apply -f lena-was-deployment-sample.yaml
```

- Deployment Result Verification

The deployed workload can be verified by executing the `kubectl get` command.

```
$ kubectl get deployments
NAME READY AGE
lena-was 2/2 10s
```

Service

The following is the Manifest file for deploying Application Service:

LENA Embedded WAS Service Specification (Manifest) File Example

```
---
apiVersion: v1
kind: Service
metadata:
  name: lena-was
spec:
  selector:
    type: lena-was
ports:
  - port: 8180
    targetPort: 8180
```

- Deployment Execution

Deployment is executed with the `kubectl apply` command. If the filename is `lena-was-service-sample.yaml`, the deployment command is as follows:

```
$ kubectl apply -f lena-was-service-sample.yaml
```

- Deployment Result Verification

The deployed workload can be verified by executing the `kubectl get` command.

```
$ kubectl get services
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
...
lena-was ClusterIP 10.43.xx.xx <none> 8180/TCP 20s
```

4.5.4. Server Registration Verification

Access Manager and select the corresponding Service Cluster from the 'Cluster > Service Cluster' menu to verify proper registration. Click the **Server List button** to verify that 2 WAS are displayed at the bottom.



Servers are registered by Manager's Scheduler, so they are automatically registered within a maximum of 15 seconds.

4.6. Web Server Deployment

4.6.1. Deployment Preparation

To register Web Server with Manager before deployment, you must register a 'Service Cluster' in Manager. Create a new Service Cluster of WEB server type in the Manager's 'Cluster > Server Cluster' menu location.

Configure Web Server settings in the Template tab of the created Service Cluster. For the integration between the previously created WAS and Web Server in the configuration content, Proxy settings are required in the Virtual Host tab.

After saving the configuration, create a Revision for the Template. A Revision must be created to enable download of the saved configuration.

For detailed information on Service Cluster creation and management, refer to the separately provided 'Administrator Manual' document.

4.6.2. Configuration Items

Basic Configuration Items

Web Server Container should be deployed based on the following recommended settings:

Configuration Item	Configuration Value / Description	Notes
Workload Type	Deployment	
Container Port	TCP : 5180	Cannot be changed

Application-Time Decision Items – Workload Related

Configuration elements that must be determined and applied according to the project environment at application time, and sample values applied to the Manifest file described later are as follows:

Configuration Item	Description	Sample Value
Container Image	LENA Manager Image matching OS and JDK version selected according to project-specific architecture decisions	lenacloud/lena-web:{TAG_NAME}
namespace	Logical group name within Kubernetes	default
name	Name of the workload, this value is used as the prefix for Pod name / Hostname	lena-web
label	Label used for connection with Service and search, defined as Key: Value pairs	type: lena-web
strategy:type	Deployment Update policy	RollingUpdate
imagePullPolicy	Image pull policy	Always
replica count	Container (Pod) count	2
Probe (Health Check)	HttpGetAction method, check page, start delay time, and period need to be set according to Application characteristics	'/' call, 5 second period, 15/20 second delay
Other Environment Variables	Environment variables can be set via ENV or Config Map	(configMap method)

The configurable environment variables are as follows. For detailed descriptions of each environment variable, refer to the [Web Server Environment Variables](#) section of this document.

Environment Variable	Description	Sample Value
LENA_CONFIG_TEMPLATE_ID	• Service Cluster name : Revision No	WEB-001:1
LENA_MANAGER_ADDRESS	• Service Domain address : port (Service address of previously installed Manager)	lena-manager.default.svc.cluster.local:7700
LENA_MANAGER_KEY	• LENA_MANAGER_KEY : Authentication token required for Manager access via Open API • Can be checked in Manager's Admin > Users menu	(Individual verification required in Manager)
LENA_CONFIG_TEMPLATE_DOWNLOAD	• Configuration information download status	Y
LENA_CONTRACT_CODE	• Contract code for license download • Code value provided when license is issued	(Individual code verification required)
LENA_LICENSE_DOWNLOAD_URL	• License download location	manager

Environment Variable	Description	Sample Value
LENA_RUN_AGENT	• Agent execution status	Y

Application-Time Decision Items – Service Related

Configuration Item	Description	Sample Value
namespace	Logical group name within Kubernetes	default
name	Name identifying the Service, must be unique within the namespace. This value is applied to the Service Domain address.	lena-web
type	Method for exposing Service externally, no separate configuration needed if there is no requirement to expose externally	NodePort (port 31180)

4.6.3. Manifest-based Deployment

Workload

Containers in Kubernetes are installed in Pod units, and the typical method is to install by creating a Manifest file in YAML file format that describes Kubernetes Objects.

The following is a sample Manifest file for installing Web Server, and the detailed content can be modified to suit the project environment according to the configuration item decisions described earlier.

LENA Web Workload Specification (Manifest) File Example

```
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: lena-web
spec:
  selector:
    matchLabels:
      type: lena-web
  replicas: 1
  strategy:
    type: RollingUpdate
  minReadySeconds: 10
  revisionHistoryLimit: 1
  template:
    metadata:
      labels:
        type: lena-web
```

```

spec:
  containers:
  - name: lena-web
    image: docker.io/lenacloud/lena-web:{TAG_NAME}
    imagePullPolicy: Always
    ports:
    - containerPort: 7180
    readinessProbe:
      httpGet:
        path: /
        port: 7180
        initialDelaySeconds: 5
        periodSeconds: 5
    livenessProbe:
      httpGet:
        path: /
        port: 7180
        initialDelaySeconds: 10
        periodSeconds: 10
    envFrom:
    - configMapRef:
        name: configmap-lena-web
    terminationGracePeriodSeconds: 0

---
apiVersion: v1
kind: ConfigMap
metadata:
  name: configmap-lena-web
data:
  LENA_MANAGER_ADDRESS: "lena-manager.default.svc.cluster.local:7700"
  LENA_AGENT_RUN: "Y"
  LENA_CONFIG_TEMPLATE_ID: "WEB-001:1"
  LENA_CONFIG_TEMPLATE_DOWNLOAD: "Y"
  LENA_MANAGER_KEY:
"aSw7RMPSw15LeN%2FMZnrxzjgV0BzZe18iVHZbJ8CkdLlea2Ecd8A1eK9oPCLXuW%3D%3D"
  LENA_LICENSE_DOWNLOAD_URL: "manager"
  LENA_CONTRACT_CODE: "dX89RRxPk6/PBPqbUuYm7w=="

```

- Deployment Execution

Deployment is executed with the `kubectl apply` command. If the filename is `lena-web-deployment-sample.yaml`, the deployment command is as follows:

```
$ kubectl apply -f lena-web-deployment-sample.yaml
```

- Deployment Result Verification

The deployed workload can be verified by executing the `kubectl get` command.


```
$ kubectl get deployments
NAME READY AGE
lena-web 2/2 10m
lena-was 2/2 10s
```

Service

The following is the Manifest file for deploying Application Service:

LENA Web Service Specification (Manifest) File Example

```
---
apiVersion: v1
kind: Service
metadata:
  name: lena-web
spec:
  selector:
    type: lena-web
  ports:
  - nodePort: 31180
    port: 7180
    targetPort: 7180
    type: NodePort
```

- Deployment Execution
Deployment is executed with the kubectl apply command. If the filename is lena-web-service-sample.yaml, the deployment command is as follows:

```
$ kubectl apply -f lena-web-service-sample.yaml
```

- Deployment Result Verification
The deployed workload can be verified by executing the kubectl get command.

```
$ kubectl get services
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
...
lena-web NodePort 10.43.xx.xx <none> 7180:31180/TCP 13h
...
```

4.6.4. Server Registration Verification

Access Manager and select the corresponding Service Cluster from the 'Cluster > Service Cluster' menu to verify proper registration. Click the **Server List button** to verify that 2 Web Servers are displayed at the bottom.



Servers are registered by Manager's Scheduler, so they are automatically registered within a maximum of 15 seconds.

Chapter 5. ECS-based Installation

This chapter explains how to deploy LENA servers or LENA image-based applications as containers in an ECS environment. It describes the installation method through the AWS Console. It also covers only the content necessary to run LENA in an ECS environment, excluding general ECS configuration parts.

5.1. ECS Overview

ECS is a container service platform provided by AWS that allows you to deploy and operate containers on EC2 instances using Docker, and group containers providing the same service into Task units to manage Replica, Service Discovery, Load Balancer, Auto-scale policies, and other features.

5.2. Installation Preparation

The following ECS environment must be prepared in advance.

- ECS Cluster and ECS task permissions
- Docker Registry accessible from ECS environment (e.g., ECR, Docker Hub)

Additionally, external storage is required to install a container-based Manager, which is used to persistently maintain existing data and files even when Manager containers are created or destroyed.

- Verify external storage environment such as EFS

For installing container-based Manager and Session Server, it is necessary to check in advance whether the environment supports Service Discovery functionality (awsvpc-based EC2, Fargate, and supported regions).

- ECS Service Discovery compatible environment

5.3. Manager Deployment

To ensure persistence of Manager instances in an ECS environment, two things are needed: 1) Fixed address 2) External volume. Fixed addresses can be achieved through ECS Service Discovery or ELB connections, and external volumes can be provided through EFS connections. The following describes Manager deployment using Service Discovery and EFS.

5.3.1. Configuration Items

Basic Configuration Items

Manager Container should be deployed based on the following recommended settings.

Table 2. ECS-based Manager Installation - Deployment Standards

Configuration Item	Setting Value / Description	Remarks
Service Type	Replica	-
Replica Count	1	-
Service Discovery	Use Service Discovery	-

Configuration Item	Setting Value / Description	Remarks
Volume Mount	Connect directory /usr/local/lena/repository to EFS	-

Decision Items for Application Timing

Configuration elements that should be determined and applied according to the project environment at the time of application, and sample values used in the installation process described later are as follows.

Table 3. ECS-based Manager Installation - Decision Items for Application Timing

Configuration Item	Description	Sample Value
Container Image	LENA Manager Image matching OS and JDK version selected according to project-specific architecture decisions	lenacloud/lena-manager:{TAG_NAME}
Probe (Health Check)	HttpGetAction, call '/lena' page	
Task and Service name	Name of Task and Service	lena-manager
Service Namespace	Used as suffix for domain address during service search	local
Service Discovery Name	Used as prefix for domain address during service search	lena-manager

Configurable environment variables are as follows. For detailed descriptions of each environment variable, refer to the [Manager Environment Variables](#) section of this document.

Table 4. ECS-based Manager Installation - Environment Variables

Environment Variable	Description	Sample Value
LENA_JVM_HEAP_SIZE	• Specify Heap Memory size	1024m (default)
LENA_JVM_METASPACE_SIZE	• Metaspace Memory size	256m (default)
LENA_MANAGER_DOMAIN_ENABLED	• Domain Name activation status	Y
LENA_MANAGER_ADDRESS	• Service Domain address : port	lena-manager.local:7700
JAVA_DOMAIN_CACHE_TTL	• Domain address Cache time (seconds)	3 (default)

5.3.2. Task Configuration

It is recommended to enter task names and permissions according to project standards. The following describes only the parts necessary for LENA operation among container definitions. The configuration

standards described below refer to the [Configuration Items](#) section.

Task Definition

Define a Task that includes container information.

작업 정의 이름* lena-manager ⓘ

호환성 요구 사항* FARGATE

작업 역할 ecsTaskExecutionRole ⓘ

인증된 AWS 서비스에 API 요청을 할 때 작업이 사용할 수 있는 IAM 역할 옵션입니다. [IAM 콘솔](#)에서 Amazon Elastic Container Service 작업 역할을 생성하십시오. [↗](#)

네트워크 모드 awsvpc ⓘ

Figure 15. ECS-based Manager Installation - Task Definition

Volume Addition

Add EFS information and storage location for storing Manager Repository in Task definition.

이름 EFS-lena-manager ⓘ

볼륨 유형 EFS ⓘ

파일 시스템 ID ⓘ

Create an Amazon Elastic File System in the [Amazon EFS Console](#) [↗](#)

Access point ID None ⓘ

Create an access point for your file system in the [Amazon EFS Console](#) [↗](#)

루트 디렉터리 /lena-manager-repository ⓘ

Figure 16. ECS-based Manager Installation - Volume Addition

Container Addition

Enter basic container information such as name and image.

컨테이너 이름* lena-manager ⓘ

이미지* docker.io/lenasupport/lena-manager:1.3.1.0_3-centos7-jdk8-openjdk ⓘ

Figure 17. ECS-based Manager Installation - Container Addition

Environment Variable Configuration

Add environment variables in container configuration. Since the Manager cannot recognize the Service Discovery address internally, enter it as the LENA_MANAGER_ADDRESS environment variable.

Figure 18. ECS-based Manager Installation - Environment Variable Configuration

Health Check Configuration

Enter Health Check information that calls the <http://localhost:7700/lena/> page.

Volume Mapping

Map the previously added volume with the internal directory of the container.

Figure 19. ECS-based Manager Installation - Volume Mapping

5.3.3. Service Configuration

Service Definition

Define a service to actually start and operate the previously defined Task.

Figure 20. ECS-based Manager Installation - Service Definition

Service Discovery Configuration

AWS supports Service Discovery functionality to connect and use services between ECS. Manager uses Service Discovery functionality to secure a fixed address and provide configuration management and monitoring functions for other servers.

Figure 21. ECS-based Manager Installation - Service Discovery Configuration

5.3.4. Service Startup and Verification

Service startup begins when the service definition is saved. Check the status of running services and tasks in the services and tasks tab of the ECS Cluster screen.

Service Status Verification

Figure 22. ECS-based Manager Installation - Service Status Verification

Task Status Verification

Figure 23. ECS-based Manager Installation - Task Status Verification

5.4. Session Server Deployment

In an ECS environment, a fixed address is also needed for Session Server services. Fixed addresses can be achieved through ECS Service Discovery or ELB connections. The following describes Session Server deployment using Service Discovery.

5.4.1. Deployment Preparation

Refer to the [Session Server Deployment Preparation](#) section of this document.

5.4.2. Configuration Items

Basic Configuration Items

Session Server Container should be deployed based on the following recommended settings.

Table 5. ECS-based Session Server Deployment Standards

Configuration Item	Setting Value / Description	Remarks
Service Type	Replica	-
Service / Replica Count	2 Services, 1 Replica per Service	-
Container Port	TCP : 5180	-
Probe	Call \${LENA_SERVER_HOME}/health.sh	-

Decision Items for Application Timing

Configuration elements that should be determined and applied according to the project environment at the time of application, and sample values applied to Task and Service configurations described later are as follows.

Table 6. ECS-based Session Server Installation - Decision Items for Application Timing

Configuration Item	Description	Sample Value
Container Image	LENA Manager Image matching OS and JDK version selected according to project-specific architecture decisions	lenacloud/lena-session:{TAG_NAME}
Task and Service name	Name of Task and Service	lena-session
Service Namespace	Used as suffix for domain address during service search	local
Service Discovery Name	Used as prefix for domain address during service search	lena-session

Configurable environment variables are as follows. For detailed descriptions of each environment variable, refer to the [Session Server Environment Variables](#) section of this document.

Table 7. ECS-based Session Server Installation - Environment Variables

Environment Variable	Description	Sample Value
LENA_JVM_HEAP_SIZE	• Specify Heap Memory size	1024m (default)
LENA_CONFIG_TEMPLATE_ID	• Service Cluster name : Revision No	SESSION-001
LENA_MANAGER_ADDRESS	• Service Domain address : port	lena-manager.local:7700
JAVA_DOMAIN_CACHE_TTL	• Domain address Cache time (seconds)	0 (default)
LENA_SESSION_0_ADDRESS	• Primary Session server service address, must match domain address according to service name and Service Discovery configuration	lena-session-0.local:5180
LENA_SESSION_1_ADDRESS	• Secondary Session server service address, must match domain address according to service name and Service Discovery configuration	lena-session-1.local:5180
LENA_SESSION_EXPIRE_SEC	• Session expiration time (seconds)	1800 (default)
LENA_CONFIG_SHARE_SESSION	• Whether to share sessions between applications	N

5.4.3. Task Configuration

It is recommended to enter task names and permissions according to project standards. The following describes only the parts necessary for LENA operation among container definitions. The configuration standards described below refer to the [Configuration Items](#) section.

Task Definition

Define a Task that includes container information.

작업 정의 이름* lena-session ⓘ

호환성 요구 사항* FARGATE

작업 역할 ecsTaskExecutionRole ⓘ

인증된 AWS 서비스에 API 요청을 할 때 작업이 사용할 수 있는 IAM 역할 옵션입니다. [IAM 콘솔](#)에서 Amazon Elastic Container Service 작업 역할을 생성하십시오. [🔗](#)

네트워크 모드 awsvpc ⓘ

Figure 24. ECS-based Session Server Installation - Task Definition

Container Addition

Enter basic container information such as name and image.

Figure 25. ECS-based Session Server Installation - Container Addition

Environment Variable Configuration

Add environment variables in container configuration.

Key	Value	
LENA_SESSION_0_ADDRESS	Value	lena-session0.local:5180 X
LENA_SESSION_1_ADDRESS	Value	lena-session1.local:5180 X
LENA_MANAGER_ADDRESS	Value	lena-manager.local:7700 X
LENA_SESSION_EXPIRE_SEC	Value	1800 X
LENA_CONFIG_TEMPLATE_ID	Value	SESSION-001 X
LENA_CONFIG_SHARE_SESSION	Value	N X
키 추가	Value	값 추가

Figure 26. ECS-based Session Server Installation - Environment Variable Configuration



The LENA_SESSION_1_ADDRESS environment variable must contain the Service Discovery address of the Secondary Session Service.

5.4.4. Service Configuration

Service Definition

Define a service to actually start and operate the previously defined Task.

작업 정의 패밀리: lana-session

버전: 1 (latest)

플랫폼 버전: LATEST

클러스터: lana-cluster-fargate-ecs

서비스 이름: lana-session0

서비스 유형*: REPLICA

작업 개수: 1

Figure 27. ECS-based Session Server Installation - Service Definition

Service Discovery Configuration

AWS supports Service Discovery functionality to connect and use services between ECS. Manager uses Service Discovery functionality to secure a fixed address and provide configuration management and monitoring functions for other servers.

서비스 검색 (선택 사항)

서비스 검색은 DNS를 통해 찾을 수 있는 네임스페이스를 생성하기 위해 Amazon Route 53을 사용합니다.

서비스 검색 통합 활성화 ☒

네임스페이스*: local | 프라이빗

서비스 검색 서비스 구성:

☒ 새로운 서비스 검색 서비스 생성

☐ 기존 서비스 검색 서비스 선택

서비스 검색 이름*: lana-session0

Figure 28. ECS-based Session Server Installation - Service Discovery Configuration

5.4.5. Service Startup and Verification

Service startup begins when the service definition is saved. Check the status of running services and tasks in the services and tasks tab of the ECS Cluster screen.

Service Status Verification

서비스	작업	ECS 인스턴스	측정치	예약된 작업	Tags	용량 공급자
<div> <div>생성</div> <div>업데이트</div> <div>삭제</div> <div>작업 ▼</div> </div>						
<div> <div>session</div> <div>시작 유형 ALL ▼</div> <div>서비스 유형 ALL ▼</div> </div>						
<input type="checkbox"/>	서비스 이름	상태	서비스 유형	작업 정의	원하는 작업 ▼	실행 중인 작업 ▼
<input type="checkbox"/>	lena-session0	ACTIVE	REPLICA	lena-session:1	1	1
<input type="checkbox"/>	lena-session1	ACTIVE	REPLICA	lena-session:1	1	1

Figure 29. ECS-based Session Server Installation - Service Status Verification

Task Status Verification

서비스	작업	ECS 인스턴스	측정치	예약된 작업	Tags	용량 공급자
<div> <div>새 작업 실행</div> <div>중지</div> <div>모두 중지</div> <div>작업 ▼</div> </div>						
원하는 작업 상태: Running Stopped						
<div> <div>session</div> <div>시작 유형 ALL ▼</div> </div>						
<input type="checkbox"/>	작업	작업 정의	컨테이너 인...	마지막 상태		
<input type="checkbox"/>	ce8a2fef-b9c0-4325-b22...	lena-session:1	--	RUNNING		
<input type="checkbox"/>	d560e17d-3f6c-471c-bec...	lena-session:1	--	RUNNING		

Figure 30. ECS-based Session Server Installation - Task Status Verification

5.5. WAS Deployment

5.5.1. Deployment Preparation

Refer to the [WAS Deployment Preparation](#) section of this document.

5.5.2. Configuration Items

Basic Configuration Items

WAS Container should be deployed based on the following recommended settings.

Table 8. ECS-based WAS Installation - Deployment Standards

Configuration Item	Setting Value / Description	Remarks
Service Type	Replica	
Container Port	TCP : 8180	-

Decision Items for Application Timing

Configuration elements that should be determined and applied according to the project environment at the time of application, and sample values applied to Task and Service configurations described later are as follows.

Table 9. ECS-based WAS Installation - Decision Items for Application Timing

Configuration Item	Description	Sample Value
Container Image	LENA Manager Image matching OS and JDK version selected according to project-specific architecture decisions	lenacloud/lena-cluster:{TAG_NAME}
Task and Service name	Name of Task and Service	lena-was
label	Label used for service connection and search, defined as Key: Value pairs	type: lena-was
Service Namespace	Used as suffix for domain address during service search	local
Service Discovery Name	Used as prefix for domain address during service search	lena-was
Probe (Health Check)	Check page, start delay time, and cycle should be set according to application characteristics	Call '/'

Configurable environment variables are as follows. For detailed descriptions of each environment variable, refer to the [WAS Environment Variables](#) section of this document.

Table 10. ECS-based WAS Installation - Environment Variables

Environment Variable	Description	Sample Value
LENA_CONFIG_TEMPLATE_ID	• Service Cluster name : Revision No	WAS-001:1
LENA_MANAGER_ADDRESS	• Service Domain address : port (Service address of previously installed Manager)	lena-manager.local:7700
LENA_MANAGER_KEY	<ul style="list-style-type: none"> • LENA_MANAGER_KEY : Authentication token required for Manager access via Open API • Can be checked in Manager's Admin > Users menu 	(Check and enter in individual Manager's Administration > IAM > Users menu)
LENA_CONFIG_TEMPLATE_DOWNLOAD	• Whether to download configuration information	Y
LENA_CONTRACT_CODE	<ul style="list-style-type: none"> • Contract code for license download • Code value provided when license is issued 	(Individual code verification required)
LENA_LICENSE_DOWNLOAD_URL		manager
JAVA_DOMAIN_CACHE_TTL	• Domain address Cache time (seconds)	3 (default)

5.5.3. Task Configuration

It is recommended to enter task names and permissions according to project standards. The following

describes only the parts necessary for LENA operation among container definitions. The configuration standards described below refer to the [Configuration Items](#) section of this document.

Task Definition

ECS-based WAS Installation - Task Definition

Define a Task that includes container information.

작업 정의 이름* lena-was ⓘ

호환성 요구 사항* FARGATE

작업 역할 ecsTaskExecutionRole ⓘ

인증된 AWS 서비스에 API 요청을 할 때 작업이 사용할 수 있는 IAM 역할 옵션입니다. [IAM 콘솔](#)에서 Amazon Elastic Container Service 작업 역할을 생성하십시오. [🔗](#)

네트워크 모드 awsvpc ⓘ

Container Addition

Enter basic container information such as name and image.

컨테이너 이름* lena-was ⓘ

이미지* docker.io/lenasupport/lena-cluster:1.3.1n.0_3-centos7-jdk8-openjdk ⓘ

Figure 31. ECS-based WAS Installation - Container Addition

Environment Variable Configuration

Add environment variables in container configuration.

환경 변수

'valueFrom' 필드를 사용하여 AWS Systems Manager 파라미터 저장소 키 또는 ARN을 지정할 수도 있습니다. ECS는 이 값을 실시간으로 컨테이너에 주입합니다.

Key	Value	
JAVA_DOMAIN_CACHE_TTL	3	✕
LENA_CONFIG_TEMPLATE_DOWNLOAD	Y	✕
LENA_CONFIG_TEMPLATE_ID	WAS-001	✕
LENA_CONTRACT_CODE	dhxq+Mjkg8C+jtv9LTkaBQ==	✕
LENA_LICENSE_DOWNLOAD_URL	manager	✕
LENA_MANAGER_ADDRESS	lena-manger.local:7700	✕
LENA_MANAGER_KEY	%2FS996W2rielBho9C4ouO%2BrEF2xRZ3FvHhZTws,	✕
키 추가	값 추가	

Figure 32. ECS-based WAS Installation - Environment Variable Configuration

5.5.4. Service Configuration

Service Definition

Define a service to actually start and operate the previously defined Task.

The screenshot shows the 'Service Definition' configuration form. It includes the following fields and values:

- 작업 정의 패밀리**: lena-was (dropdown menu)
- 버전**: 7 (latest) (dropdown menu)
- 플랫폼 버전**: LATEST (dropdown menu)
- 클러스터**: lena-cluster-fargate-ecs (dropdown menu)
- 서비스 이름**: lena-was (text input)
- 서비스 유형***: REPLICA (radio button)
- 작업 개수**: 1 (text input)

There is a '값 입력' (Enter value) button next to the '작업 정의 패밀리' dropdown. Information icons (i) are present next to the '플랫폼 버전', '클러스터', '서비스 이름', '서비스 유형*', and '작업 개수' fields.

Figure 33. ECS-based WAS Installation - Service Definition

Service Discovery Configuration

AWS supports Service Discovery functionality to connect and use services between ECS. Manager uses Service Discovery functionality to secure a fixed address and provide configuration management and monitoring functions for other servers.

The screenshot shows the 'Service Discovery Configuration' form. It includes the following fields and values:

- 서비스 검색 (선택 사항)**: 서비스 검색은 DNS를 통해 찾을 수 있는 네임스페이스를 생성하기 위해 Amazon Route 53을 사용합니다.
- 서비스 검색 통합 활성화**: ☒ (checkbox)
- 네임스페이스***: local | 프라이빗 (dropdown menu)
- 서비스 검색 서비스 구성**:
 - ☒ 새로운 서비스 검색 서비스 생성
 - ☐ 기존 서비스 검색 서비스 선택
- 서비스 검색 이름***: lena-was (text input)

Information icons (i) are present next to the '네임스페이스*' and '서비스 검색 이름*' fields.

Figure 34. ECS-based WAS Installation - Service Discovery Configuration

5.5.5. Service Startup and Verification

Service startup begins when the service definition is saved. Check the status of running services and tasks in the services and tasks tab of the ECS Cluster screen.

Service Status Verification

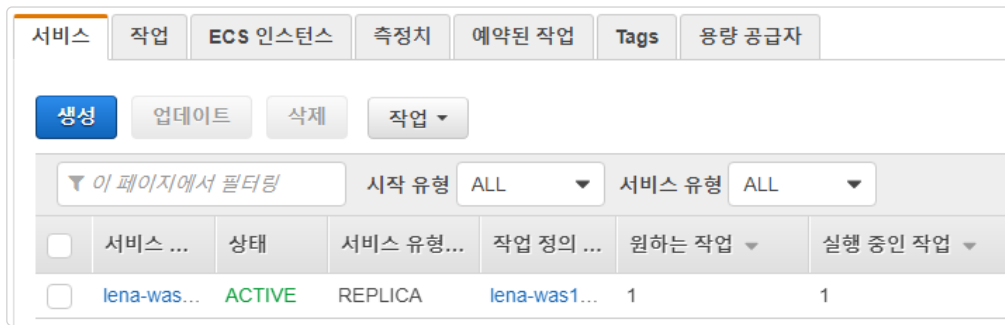


Figure 35. ECS-based WAS Installation - Service Status Verification

Task Status Verification

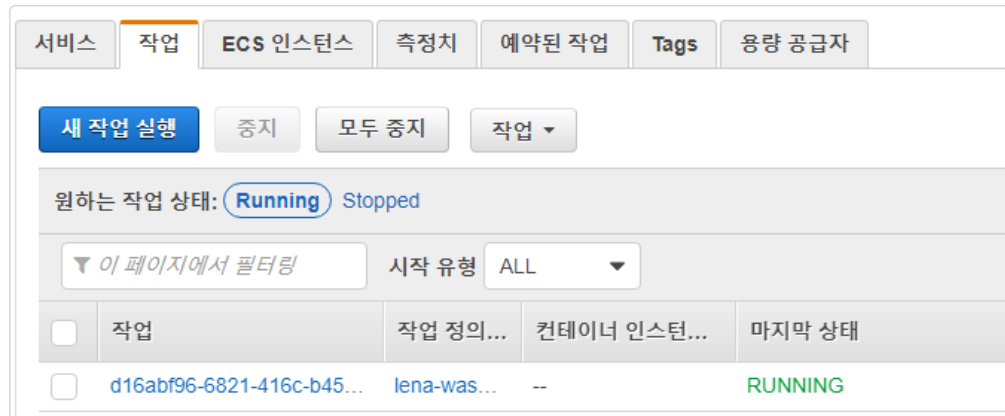


Figure 36. ECS-based WAS Installation - Task Status Verification

5.6. Embedded WAS Deployment

5.6.1. Deployment Preparation

Refer to the [Embedded WAS Deployment Preparation](#) section of this document.

5.6.2. Configuration Items

Basic Configuration Items

Embedded WAS Container should be deployed based on the following recommended settings.

Table 11. ECS-based Embedded WAS Installation - Deployment Standards

Configuration Item	Setting Value / Description	Remarks
Service Type	Replica	
Container Port	TCP : 8180	-

Decision Items for Application Timing

Configuration elements that should be determined and applied according to the project environment at the time of application, and sample values applied to Task and Service configurations described later are as follows.

Table 12. ECS-based WAS Installation - Decision Items for Application Timing

Configuration Item	Description	Sample Value
Container Image	LENA Manager Image matching OS and JDK version selected according to project-specific architecture decisions	lenacloud/lena-embedded:{TAG_NAME}
Task and Service name	Name of Task and Service	lena-was
label	Label used for service connection and search, defined as Key: Value pairs	type: lena-was
Service Namespace	Used as suffix for domain address during service search	local
Service Discovery Name	Used as prefix for domain address during service search	lena-was
Probe (Health Check)	Check page, start delay time, and cycle should be set according to application characteristics	Call '/'

Configurable environment variables are as follows. For detailed descriptions of each environment variable, refer to the [Embedded WAS Environment Variables](#) section of this document.

Environment Variable	Description	Sample Value
LENA_CONFIG_TEMPLATE_ID	• Service Cluster name : Revision No	WAS-001:1
LENA_MANAGER_ADDRESS	• Service Domain address : port (Service address of previously installed Manager)	lena-manager.default.svc.cluster.local:7700
LENA_MANAGER_MONITORING_PORT	• Manager monitoring Port information	16100
LENA_APP_FILE	• Application Jar file name	sample-app.jar
LENA_APP_DIR	• Application Jar directory name	/usr/local/lena

5.6.3. Task Configuration

It is recommended to enter task names and permissions according to project standards. The following describes only the parts necessary for LENA operation among container definitions. The configuration standards described below refer to the [Configuration Items](#) section of this document.

Task Definition

ECS-based Embedded WAS Installation - Task Definition

Define a Task that includes container information.

작업 정의 이름* lena-was ⓘ

호환성 요구 사항* FARGATE

작업 역할 ecsTaskExecutionRole ⓘ

인증된 AWS 서비스에 API 요청을 할 때 작업이 사용할 수 있는 IAM 역할 옵션입니다. [IAM 콘솔](#)에서 Amazon Elastic Container Service 작업 역할을 생성하십시오. [🔗](#)

네트워크 모드 awsvpc ⓘ

Container Addition

Enter basic container information such as name and image.

컨테이너 이름* lena-was ⓘ

이미지* docker.io/lenasupport/lena-cluster:1.3.1n.0_3-centos7-jdk8-openjdk ⓘ

Figure 37. ECS-based Embedded WAS Installation - Container Addition

Environment Variable Configuration

Add environment variables in container configuration.

환경 변수

'valueFrom' 필드를 사용하여 AWS Systems Manager 파라미터 저장소 키 또는 ARN을 지정할 수도 있습니다. ECS는 이 값을 실시간으로 컨테이너에 주입합니다.

Key	Value	
JAVA_DOMAIN_CACHE_TTL	Value ▼ 3	✕
LENA_CONFIG_TEMPLATE_DOWNLOAD	Value ▼ Y	✕
LENA_CONFIG_TEMPLATE_ID	Value ▼ WAS-001	✕
LENA_CONTRACT_CODE	Value ▼ dhxq+Mjkg8C+jtv9LTkaBQ==	✕
LENA_LICENSE_DOWNLOAD_URL	Value ▼ manager	✕
LENA_MANAGER_ADDRESS	Value ▼ lena-manger.local:7700	✕
LENA_MANAGER_KEY	Value ▼ %2FS996W2rielBho9C4ouO%2BrEF2xRZ3FvHhZTws,	✕
키 추가	Value ▼	값 추가

Figure 38. ECS-based Embedded WAS Installation - Environment Variable Configuration

5.6.4. Service Configuration

Service Definition

Define a service to actually start and operate the previously defined Task.

Figure 39. ECS-based WAS Installation - Service Definition

Service Discovery Configuration

AWS supports Service Discovery functionality to connect and use services between ECS. Manager uses Service Discovery functionality to secure a fixed address and provide configuration management and monitoring functions for other servers.

Figure 40. ECS-based Embedded WAS Installation - Service Discovery Configuration

5.6.5. Service Startup and Verification

Service startup begins when the service definition is saved. Check the status of running services and tasks in the services and tasks tab of the ECS Cluster screen.

Service Status Verification

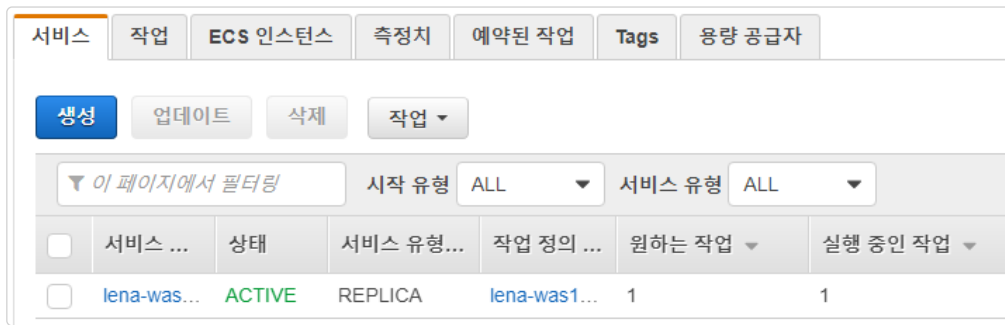


Figure 41. ECS-based WAS Installation - Service Status Verification

Task Status Verification

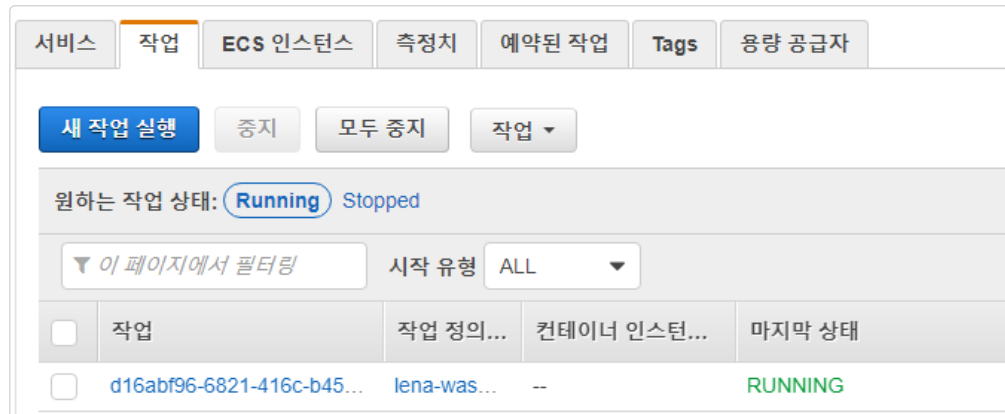


Figure 42. ECS-based WAS Installation - Task Status Verification

5.7. Web Server Deployment

5.7.1. Deployment Preparation

Refer to the [Web Server Deployment Preparation](#) section of this document.

5.7.2. Configuration Items

Basic Configuration Items

Web Server Container should be deployed based on the following recommended settings.

Table 13. ECS-based Web Server Installation - Deployment Standards

Configuration Item	Setting Value / Description	Remarks
Service Type	Replica	-
Container Port	TCP : 7180	-

Decision Items for Application Timing

Configuration elements that should be determined and applied according to the project environment at the time of application, and sample values applied to ECS Task and Service configurations described later are as follows.

Table 14. ECS-based Web Server Installation - Decision Items for Application Timing

Configuration Item	Description	Sample Value
Container Image	LENA Manager Image matching OS and JDK version selected according to project-specific architecture decisions	lenacloud/lena-web:{TAG_NAME}
Task and Service name	Task name, this value is used as prefix for Task name / Hostname	lena-web
replica count	Number of Containers (Tasks)	2
Probe (Health Check)	Start delay time and cycle should be set according to application characteristics	Call '/'
Service Namespace	Used as suffix for domain address during service search	local
Service Discovery Name	Used as prefix for domain address during service search	lena-web

Configurable environment variables are as follows. For detailed descriptions of each environment variable, refer to the [Web Server Environment Variables](#) section of this document.

Table 15. ECS-based Web Server Installation - Environment Variables

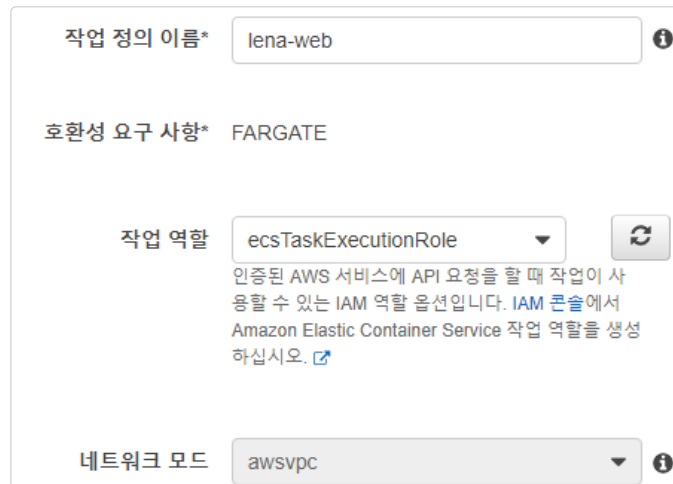
Environment Variable	Description	Sample Value
LENA_CONFIG_TEMPLATE_ID	<ul style="list-style-type: none"> Service Cluster name : Revision No Downloads Default Revision if Revision No is empty 	WEB-001:1
LENA_MANAGER_ADDRESS	<ul style="list-style-type: none"> Service Domain address : port (Service address of previously installed Manager) 	lena-manager.local:7700
LENA_MANAGER_KEY	<ul style="list-style-type: none"> LENA_MANAGER_KEY : Authentication token required for Manager access via Open API Can be checked in Manager's Admin > Users menu 	(Individual verification and input required in each Manager)
LENA_CONFIG_TEMPLATE_DOWNLOAD	<ul style="list-style-type: none"> Whether to download configuration information 	Y
LENA_CONTRACT_CODE	<ul style="list-style-type: none"> Contract code for license download Code value provided when license is issued 	(Individual code verification required)
LENA_LICENSE_DOWNLOAD_URL	<ul style="list-style-type: none"> License download location 	manager
LENA_RUN_AGENT	<ul style="list-style-type: none"> Whether to run Agent 	Y

5.7.3. Task Configuration

Enter task names and permissions according to project standards, and describe only the parts necessary for LENA operation among container definitions. The configuration standards described below refer to the [Configuration Items](#) section of this document.

Task Definition

Define a Task that includes container information.



The screenshot shows the 'Task Definition' configuration window. It includes the following fields and options:

- 작업 정의 이름*** (Task Definition Name): lena-web
- 호환성 요구 사항*** (Compatibility Requirement): FARGATE
- 작업 역할** (Task Role): ecsTaskExecutionRole. Below this, there is a note: '인증된 AWS 서비스에 API 요청을 할 때 작업이 사용할 수 있는 IAM 역할 옵션입니다. [IAM 콘솔](#)에서 Amazon Elastic Container Service 작업 역할을 생성하십시오.' (This is an IAM role option that the task can use to make API requests to authorized AWS services. Create an Amazon Elastic Container Service task role in the [IAM console](#).)
- 네트워크 모드** (Network Mode): awsvpc

Figure 43. ECS-based Web Server Installation - Task Definition

Container Addition

Enter basic container information such as name and image.



The screenshot shows the 'Container Addition' configuration window. It includes the following fields:

- 컨테이너 이름*** (Container Name): lena-web
- 이미지*** (Image): docker.io/lenasupport/lena-web:1.3.1n.0_3-centos7-jdk8-openjdk

Figure 44. ECS-based Web Server Installation - Container Addition

Environment Variable Configuration

Add environment variables in container configuration.

환경 변수

'valueFrom' 필드를 사용하여 AWS Systems Manager 파라미터 저장소 키 또는 ARN을 지정할 수도 있습니다. ECS는 이 값을 실시간으로 컨테이너에 주입합니다.

Key

LENA_AGENT_RUN	Value	Y	✕
LENA_CONFIG_TEMPLATE_DOWNLOAD	Value	Y	✕
LENA_CONFIG_TEMPLATE_ID	Value	WEB-001	✕
LENA_CONTRACT_CODE	Value	dhxq+Mjkg8C+jtv9LTkaBQ==	✕
LENA_LICENSE_DOWNLOAD_URL	Value	manager	✕
LENA_MANAGER_ADDRESS	Value	lena-manager.local:7700	✕
LENA_MANAGER_KEY	Value	%2FS996W2rieIBho9C4ouO%2BrEF2xRZ3FvZhZTws,	✕
키 추가	Value	값 추가	

Figure 45. ECS-based Web Server Installation - Environment Variable Configuration

5.7.4. Service Configuration

Service Definition

Define a service to actually start and operate the previously defined Task.

작업 정의 패밀리

lena-web

값 입력

버전

1 (latest)

클러스터

LN-TEST-01

서비스 이름

lena-web

서비스 유형*

☒ REPLICa ☐ DAEMON

작업 개수

1

Figure 46. ECS-based Web Server Installation - Service Definition

Service Discovery Configuration

Web Server can secure a fixed address using ELB or Service Discovery functionality to provide web services to external or other services.

서비스 검색 (선택 사항)

서비스 검색은 DNS를 통해 찾을 수 있는 네임스페이스를 생성하기 위해 Amazon Route 53을 사용합니다.

서비스 검색 통합 활성화 ☒

네임스페이스*

local | 프라이빗

서비스 검색 서비스 구성

☒ 새로운 서비스 검색 서비스 생성 ☐ 기존 서비스 검색 서비스 선택

서비스 검색 이름*

lena-web

Figure 47. ECS-based Web Server Installation - Service Discovery Configuration

5.7.5. Service Startup and Verification

Service startup begins when the service definition is saved. Check the status of running services and tasks in the services and tasks tab of the ECS Cluster screen.

Service Status Verification



Figure 48. ECS-based Web Server Installation - Service Status Verification

Task Status Verification

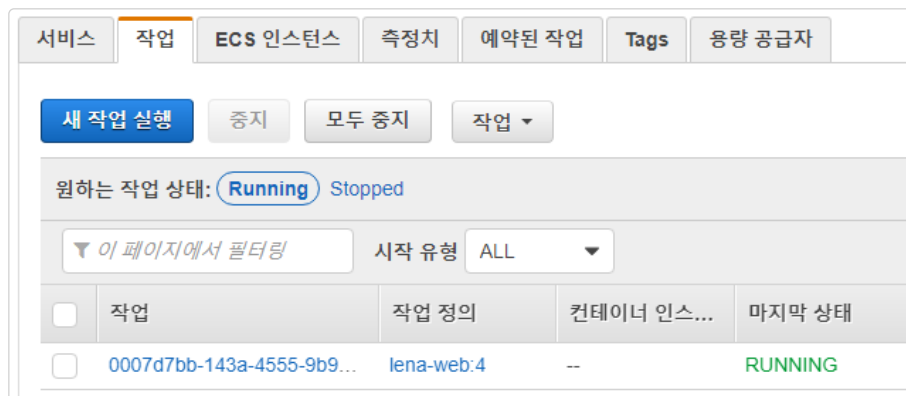


Figure 49. ECS-based Web Server Installation - Task Status Verification

Chapter 6. VM/Host-based Installation

6.1. Installation Preparation

As part of the installation preparation, upload the installation files to the target server and install and run the Manager and Node Agent. Subsequent installation tasks can be performed through the Manager's Web UI, and if the Web UI is inaccessible due to a firewall, the same installation can be performed via the command line.

6.2. LENA Installation

LENA installation files are provided in gzip format. After uploading them to the target server, extract them into the installation home directory (\${LENA_HOME}). The default installation path is '/engn001/lena/1.3/'.

LENA Installation

```
[engn001]#  
[engn001]# tar -xzvf lena-1.3.x.tar.gz
```

The installation modules are provided by purpose as follows.

Table 16. LENA Installation Module List

Scripts	Description	Remarks
lena-[version].tar.gz	Integrated installation file for Web/Application which includes Application Server, Web Server, and Session Server installation modules	lena-1.3.x.tar.gz
lena-enterprise-[version].tar.gz	WAS installation module for the Enterprise edition The Enterprise edition includes the Session Server	lena-enterprise-1.3.x.tar.gz
lena-standard-[version].tar.gz	WAS installation module for the Standard edition	lena-standard-1.3.x.tar.gz

6.3. Directory Structure

Prepare the files for LENA installation. LENA installation files are provided separately.

The directory structure of \${LENA_HOME} is as follows.

Table 17. Directory Structure

Directory	Description	Remarks
bin	Start/Stop scripts for Node Agent and Manager, provides install scripts	

Directory	Description	Remarks
conf	Configuration files for Node Agent, Manager, etc.	
database	Directory for storing daily data generated by monitoring	
depot	Local repository for installation	
etc	Other meta information and configuration files	
license	Directory for managing license information	
logs	Node Agent / Manager log files	
modules	Path where modules required for execution are located (lena-node-agent, lena-installer, lena-manager, etc.)	
servers	Default path where servers are installed	
tmp	Temporary directory	

The provided execution scripts are as follows. (Located at \${LENA_HOME}/bin)

Table 18. Provided Script List

Scripts	Description	Remarks
install.sh	Basic script for installing servers	
web-compile.sh	Script for compiling the Web Server	
web-package-install.sh	Script for installing packages required for Web Server compilation and operation	Linux only, root privileges required
crypt.sh	Manual encryption execution for the password used in the DataSource (converts the input string to an encrypted string)	
env-manager.sh	Environment variables for running the Manager	When Manager is installed
start-manager.sh	Run Manager	When Manager is installed
stop-manager.sh	Stop Manager	When Manager is installed
ps-manager.sh	Check Manager process	When Manager is installed
start-agent.sh	Run Node Agent	
stop-agent.sh	Stop Node Agent	

Scripts	Description	Remarks
ps-agent.sh	Check Node Agent process	

The configuration files are as follows. (Located at \${LENA_HOME}/conf)

Table 19. Configuration File List

Config File	Description	Remarks
manager.conf	Manager-related settings	
agent.conf	Node Agent-related settings	

6.4. Manager Installation

The provided LENA consists of Web Server, WAS and Session Server, an Agent installed on Node/Server to control and check status, and a Manager provided for administrators.

6.4.1. Manager Installation

Install the Manager using install.sh in the following order.

1. \${LENA_HOME}/bin/install.sh create lena-manager
2. Enter the Service Port information. (default: 7700)
3. Enter the port information for receiving server status information. Use the default setting, and when installing an additional Manager, change the port. (default: 16100)
4. Enter the OS account to run the Manager. (default: script execution user)

LENA Manager Installation

```
[bin]$ ./install.sh create lena-manager
*****
* LENA Server Install !          *
*****

+-----+
| 1. SERVICE_PORT is the port number used by Manager.
|    ex : 7700
| 2. MONITORING_PORT is the port number used by Manager for monitoring.
|    ex : 16100
| 3. RUN_USER is user running Argo Manager.
|    ex : tomat
+-----+
-----
Input SERVICE_PORT for installation. (q:quit)
Default value is '7700'

Input MONITORING_PORT for installation. (q:quit)
Default value is '16100'

Input RUN_USER for installation. (q:quit)
Default value is 'lena'

===== Execution Result =====
LENA_HOME   : /engn001/lena/1.3
JAVA_HOME   : /engn001/java/jdk1.8.0_191
SERVER_ID    : lena-manager
SERVICE_PORT : 7700
MONITORING_PORT : 16100
INSTALL_PATH : /engn001/lena/1.3/modules/lena-manager
RESULT       : Success
MESSAGE      : create succeeded
=====

Execution is completed.!!
[bin]$
```



When operating services across multiple machines, install the Manager on only one machine.

6.4.2. Manager Execution

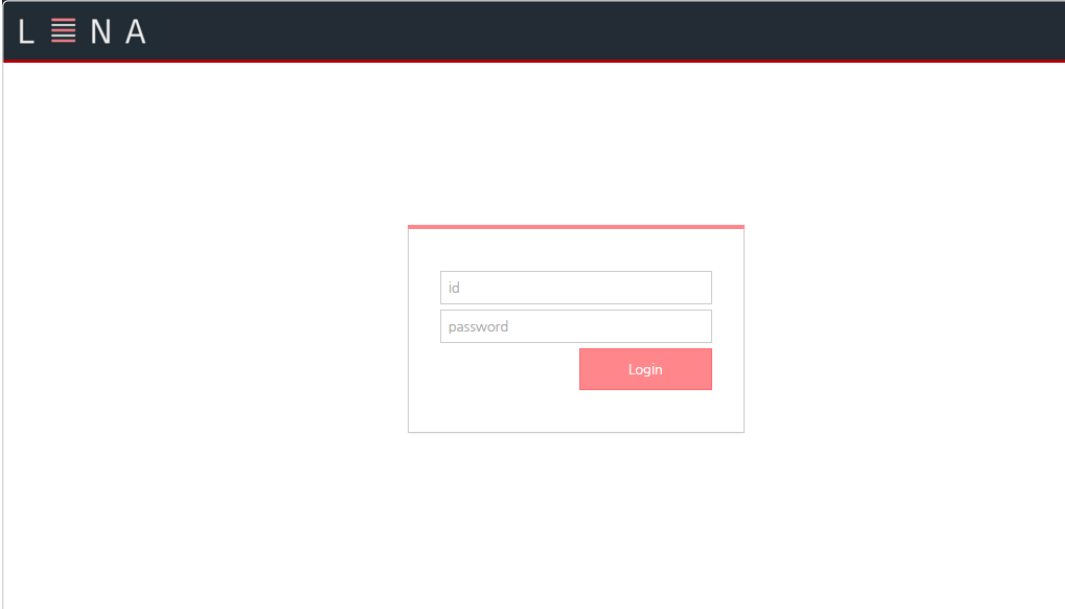
Start the Manager to verify that it has been installed correctly.

1. Run the start-manager.sh file.

LENA Manager Execution

```
[bin]$ ./start-manager.sh
-----
                LENA Manager
-----
Using LENA_HOME :    /engn001/lena/1.3
Using JRE_HOME   :    /engn001/java/jdk1.8.0_191
Using SERVER_PID :    /engn001/lena/1.3/modules/lena-manager/lena-
manager_solmanager.pid
Using SERVER_HOME : /engn001/lena/1.3/modules/lena-manager
Using SERVER_ID  :    lena-manager
Using INSTANCE_NAME : lena-manager_solmanager
LENA started.
[bin]$
```

2. Access [http://\[Manager IP\]:7700](http://[Manager IP]:7700) and check the page below. (Initial credentials: admin/!admin1234)



The screenshot shows a web browser window displaying the LENA Manager login page. The header is dark blue with the text 'L N A' and a hamburger menu icon. The main content area is white and contains a login form with two input fields labeled 'id' and 'password', and a red 'Login' button.

Figure 50. LENA Manager Login

3. Run the stop-manager.sh file to stop.

LENA Manager Stop

```
[bin]$ ./stop-manager.sh
-----
                LENA Manager
-----
Using LENA_HOME : /engn001/lena/1.3
Using JRE_HOME: /engn001/java/jdk1.8.0_191
Using SERVER_PID: /engn001/lena/1.3/modules/lena-manager/lena-
manager_solmanager.pid
Using SERVER_HOME : /engn001/lena/1.3/modules/lena-manager
Using SERVER_ID : lena-manager
Using INSTANCE_NAME : lena-manager_solmanager
LENA stopped.
##### lena-manager_solmanager successfully shut down #####
[bin]$
```

6.5. Node Agent Execution

Node Agent is the agent responsible for control and monitoring of Nodes and Servers. Node Agent is installed by default during LENA installation, and it must be started to retrieve information about the Node. Node Agent can check the status and start/stop Web/Application/Session Servers.

6.5.1. Node Agent Execution

Run the \${LENA_HOME}/bin/start-agent.sh file. If JAVA_HOME is not specified, the terminal prompts you to enter JAVA_HOME. Enter the JAVA_HOME path and the agent will start.

```
[bin]# ./start-agent.sh
Input JAVA_HOME path for LENA. ( q: quit )
JAVA_HOME PATH :
/engn001/java/jdk1.8.0_191
Input Agent port for LENA Agent. ( q: quit )
Agent port (Default : 16800):
Input Agent user for LENA Agent. ( q: quit )
Agent user (Default : root):
root

-----
                LENA Agent
-----

Using LENA_HOME : /engn001/lena/1.3
Using JAVA_HOME : /engn001/java/jdk1.8.0_191/jre
Using CONF_FILE : /engn001/lena/1.3/conf/agent.conf
Using LOG_HOME : /engn001/lena/1.3/logs/lena-agent
Using RUN_USER : root
Using PORT : 16800
Using UUID : d03ddd60-de12-35df-9ea1-a409a3085eeb
LENA Agent is started.
[bin]#
```

6.5.2. Node Agent Operation Check

Run the \${LENA_HOME}/bin/ps-agent.sh file to check the process status as shown below.

```
[bin]$ ./ps-agent.sh
lena      24208      1 62 14:00 ?        00:00:03
/engn001/java/jdk1.8.0_191/bin/java -Xms64m -Xmx256m
-Dlena.home=/engn001/lena/1.3 -Dlog.home=/engn001/lena/1.3/logs/lena-agent
-Dpatch.log.home=/engn001/lena/1.3/logs/lena-patcher
-Djava.library.path=:/engn001/lena/1.3/modules/lena-agent/lib/sigar
-Djava.net.preferIPv4Stack=true -cp ./engn001/lena/1.3/modules/lena-
agent/lib/bcprov-jdk15on-1.55.jar:/engn001/lena/1.3/modules/lena-
agent/lib/lena-agent-1.3.0.jar:/engn001/lena/1.3/modules/lena-
agent/lib:/engn001/java/jdk1.8.0_191/lib/tools.jar
argo.node.agent.server.NodeAgentServer -start
[bin]$
```

6.5.3. Node Agent Stop

Run stop-agent.sh to stop.

```
[bin]$ ./stop-agent.sh

-----

      LENA Agent

-----

Using LENA_HOME : /engn001/lena/1.3
Using JAVA_HOME : /engn001/java/jdk1.8.0_191/jre
Using CONF_FILE : /engn001/lena/1.3/conf/agent.conf
Using LOG_HOME  : /engn001/lena/1.3/logs/lena-agent
Using RUN_USER  : lena
Using PORT      : 16800
Using UUID      : 0d5f6a4a-1084-4bac-ad8c-70b67bf3e495
LENA Agent is stopped normally.
[bin]$
```

6.6. Session Server Installation (WEB UI-based)

A screen is provided to manage the Session Server. Registration, modification, and deletion of Session Servers installed on the Node are possible, and start and stop shells can be executed.

Status	Name	IP	Server ID	Port	Server Type
OK	tm-session1_5105	10.0.1.88	tm-session1_5105	5105	Standalone
OK	tm-session2_5106	10.0.1.88	tm-session2_5106	5106	Standalone

Figure 51. Session Server List

The properties of the Session Server are as follows. (*) indicates required values

Table 20. Session Server Properties

Item	Description	Remarks
Status	Session Server status	
Name(*)	Session Server name	
IP(*)	Session Server IP address	
Server ID	Session Server identifier	
Port	Service port number	
Server Type	Session Server type	
Start/Stop	Server start and stop	
+ icon	Click the Register button or Edit (pencil) button to indicate that the selected Server information is being modified	

Item	Description	Remarks
- icon	Click the Delete (trash) button to indicate that the selected Server information is being deleted	

6.6.1. Session Server Installation

1. Click the **Install button**.

Install

Input server information for installation.

- * Server Type: ☒ Standalone
- * Server ID:
- * Service Port:
- * Secondary Server IP:
- * Secondary Server Port:
- * Run User:
- * Manager IP:
- * Install Root Path:

☒ Save

Figure 52. Session Server Installation Input Screen

2. Enter the Server ID, Service Port, and Secondary Server IP/Port.
3. Click the 'Save' button to save.



- There may be differences between the servers actually installed on the Node and the server information managed by the Manager. (in console-based installation)
- If a duplicate server ID error occurs, use the Register function to check the installed server information.
- The Manager IP is automatically entered as the host IP of the Node. Depending on the network configuration, the automatically entered IP may be different from the actual network IP. In this case, modify and enter the Manager IP.

6.6.2. Server Start

1. Click the **Stop button** to stop the Server.
2. Click the **Start button** to start the Server.



The Start button is activated only when the server is in a startable state.

6.6.3. Server Delete

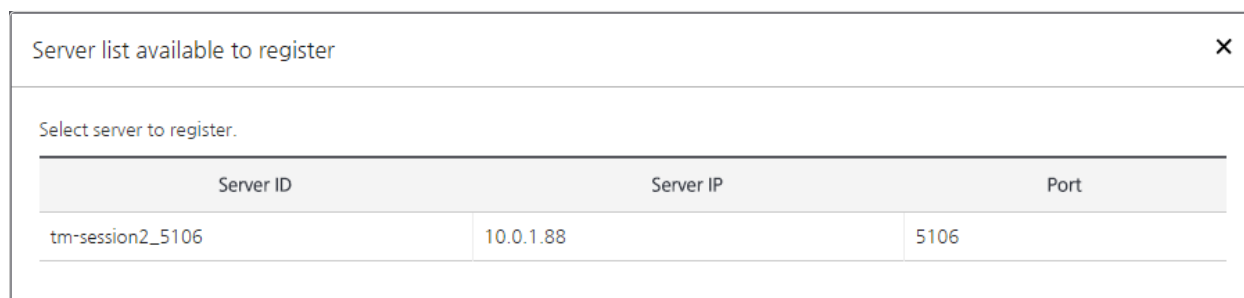
1. Click the **Delete (trash) button** to change the Server information to a deletable state.
2. Click the **Save button**.
3. Clicking OK completely deletes the Manager's DB data and the physical server, while clicking Cancel

deletes only the Manager's DB data.

6.6.4. Server Registration

To manage a server installed via console through the Manager, server information registration is required.

1. Click the **+Register button**.
2. Click the server to register.



Server list available to register

Select server to register.

Server ID	Server IP	Port
tm-session2_5106	10.0.1.88	5106

Figure 53. Session Server Registration – Server Selection Screen

3. Click the **Save button** to save.

6.7. Session Server Installation (CLI-based)

6.7.1. Session Server Installation

The Session Server is divided into Embedded and Standalone versions. The Embedded version is included within the Application Server and does not require a separate installation; for the Standalone version, install in the following order using install.sh.

1. `${LENA_HOME}/bin/install.sh create lena-session`

Session Server Installation

```
[bin]$ ./install.sh create lena-session
*****
* LENA Server Install !          *
*****

+-----+
|-----|
| 1. SERVER_ID means business code of system and its number of letter is
|    from 3 to 5.
|    ex : tom1, tc01, svr01
| 2. SERVICE_PORT is the port number used by Session Server.
|    ex : 8080
| 3. SECONDARY_SERVER_IP is the ip number communicate with Secondary Session
Server
|    ex : 127.0.0.1
| 4. SECONDARY_SERVICE_PORT is the port number used by Secondary Session
| Server.
|    ex : 8080
| 5. RUN_USER is user running Session Server
|    ex : tomat, apahe
| 6. INSTALL_ROOT_PATH is is server root directory in filesystem.
|    ex : /ssw, /sw/server, /ssw/was
+-----+
|-----|
```

2. Input Items

- The default value is shown for each item, and if changes are needed, the user can directly enter values to modify them.

Session Server Installation – Input Item Example

```

Input SERVER_ID for installation. (q:quit)
tm-session1
Input SERVICE_PORT for installation. (q:quit)
Default value is '5000'
5005
Input SECONDARY_SERVER_IP for installation. (q:quit)
127.0.0.1
Input SECONDARY_SERVICE_PORT for installation. (q:quit)
Default value is '5001'
5006
Input RUN_USER for installation. (q:quit)
Default value is 'lena'

Input INSTALL_ROOT_PATH for installation. (q:quit)
Default value is '/engn001/lena-1.3.0/tmservers'

===== Execution Result =====
LENA_HOME : /engn001/lena/1.3
JAVA_HOME : /engn001/java/jdk1.8.0_191/jre
SERVER_ID : tm-session1
SERVICE_PORT : 5005
SECONDARY_SERVER_IP : 127.0.01
SECONDARY_SERVICE_PORT : 5006
RUN_USER : lena
INSTALL_PATH : /engn001/lena/1.2/servers/session1
RESULT : Success
MESSAGE : create succeeded
=====

create is completed.!!
[bin]$

```

Table 21. Session Server Installation – Input Items

Item	Description	Remarks
SERVER_ID	Session Server ID	
SERVICE_PORT	Session Server service port	Default: "5000"
SECONDARY_SERVER_IP	Secondary Server IP address	
SECONDARY_SERVICE_PORT	Secondary Server service port	Default: "5001"
RUN_USER	Execution account name for running Session Server	Default: "script execution account"

Item	Description	Remarks
INSTALL_ROOT_PATH	Parent directory where the Session Server will be installed	Default: <code>"\${LENA_HOME}/tmse rvers"</code>

- Verify that the `$INSTALL_ROOT_PATH/tmservers/"SERVICE_ID"` directory is created.



When running `install.sh`, one Session Server is installed. To install N servers, run `install.sh` N times.

6.7.2. Session Server Start

Start the Session Server to verify that it has been installed correctly.

1. From the Session Server installation location, run the `start.sh` file.

Session Server Start

```
[tm-session1]$ ./start.sh
-----
Start Session Server
-----
Using LENA_HOME : /engn001/lena/1.3
Using SERVER_HOME : /engn001/lena/1.3/servers/tm-session1
Using SERVER_ID : tm-session1
Using JAVA_HOME : /engn001/java/jdk1.8.0_191

Session Server Started..
[tm-session1]$
```

2. Run the `ps.sh` file to check the process status.

Session Server – Process Status Check

```
[tm-session1]$ ./ps.sh

lena 16232      1      1 09:56 pts/7 00:00:00
/engn001/java/jdk1.8.0_191/bin/java -Xmx1024m -Dzodiac.name=session_5105
-Dzodiac.logdir=/engn001/lena/1.3/logs/session-server -cp
.::/engn001/lena/leesyong/1.2/servers/tm-session1/lib/lena-session-common-
1.2.0.jar:/engn001/lena/leesyong/1.2/servers/tm-session1/lib/lena-session-
server-1.2.0.jar -Dzodiac.config=session.conf zodiac.server.Main

[tm-session1]$
```

Session Server Stop

3. Run the `stop.sh` file to stop.

```
[tm-session1]$ ./stop.sh
-----
Stop Session Server
-----
Using LENA_HOME : /engn001/lena/1.3
Using SERVER_HOME : /engn001/lena/1.3/servers/tm-session1
Using SERVER_ID : tm-session1
Using JAVA_HOME : /engn001/java/jdk1.8.0_191

Session Server Stopped..
[tm-session1]$
```

6.7.3. Session Server Delete

A pre-installed server can be uninstalled using scripts.

LENA stores information about installed servers in a separate xml file. Therefore, do not delete the directory directly; instead, use the install.sh script to uninstall.

1. Run the install.sh script
 - Session Server : `${LENA_HOME}/bin/install.sh delete lena-session`
 - Manager : `${LENA_HOME}/bin/install.sh delete lena-manager`

Session Server Delete

```
[lena@RNDTOMCAT1V bin]$ ./install.sh delete tm-session
*****
* LENA Server Install !          *
*****

+-----+
| 1. SERVER_ID : Server'id to delete
+-----+

-----
Input SERVER_ID for installation. (q:quit)
tm-session
===== Execution Result =====
LENA_HOME : /engn001/lena/1.3
JAVA_HOME : /engn001/java/jdk1.8.0_191/jre
SERVER_ID : lenawas2
DELETE_PATH : /engn001/lena/1.3/servers/tm-session
RESULT : Success
MESSAGE : delete succeeded
=====

delete is completed.!!

[bin]$
```

2. Input Items

Table 22. Session Server Delete – Input Items

Item	Description	Remarks
SERVER_ID	ID of the server to uninstall	For the Manager, the id is automatically entered as lena-manager, and the Server ID is not requested separately.



LENA stores information about installed servers in a separate xml file. Therefore, do not delete the directory directly; instead, use the install.sh script to uninstall.

Chapter 7. Appendix

7.1. LENA Supported Spec Versions

Table 23. LENA Supported Spec

Specification	Version	Remarks
Java Development Kit (JDK)	1.8~	
Java Servlet	3.1	
Java Server Pages (JSP)	2.3	
Expression Language (EL)	2.2	
JavaServer Pages Standard Tag Library (JSTL)	1.2	
Enterprise JavaBeans (EJB)	3.2	
Java Message Service (JMS)	1.1	
Java Transaction API (JTA)	1.2	
Java API for RESTful Services (JAX-RS)	2.0	
Java API for XML Web Services (JAX-WS)	2.2	

7.2. Manager DB File Backup

The HSQL DB files for managing the Manager's internal data are backed up periodically (daily). The location is `${LENA_HOME}/repository/backup/database`.

By default, backup information older than 30 days is deleted. If you want to change the retention period, open the `manager.conf` file under the `${LENA_HOME}/conf` folder, enter `dbbackup.size=[retention period]`, and restart the Manager to apply the new retention period.

7.3. Deletion of Manager Internal History

The history that the Manager keeps internally is scheduled to be deleted periodically. The information deleted includes Action Trace history and Server History.

By default, Action Trace history is retained for 30 days and Server History is retained for 90 days. If you want to change these retention periods, open the `manager.conf` file under the `${LENA_HOME}/repository/conf` folder, enter `actiontrace.size=[retention period]`, `serverhistory.size=[retention period]`, and restart the Manager to apply the changes.

7.4. Manager admin Password Reset

If the Manager's admin user password is lost or the number of incorrect password attempts is exceeded, you must reset the password via the console.

1. Connect to the machine where the Manager is installed via console (telnet or ssh).
2. Run `$LENA_HOME/bin/reset_manager_pw.sh`.

3. Enter admin as the user whose password will be reset.
4. Enter the new password to initialize. The password must be at least 8 characters long and be a combination of letters/numbers/special characters. For security, the password is not displayed on the console.

Manager admin password reset

```
[bin]$ ./reset-manager-pw.sh
*****
* LENA Server Install !      *
*****

+-----+
--
| 1. USER_ID is the user id to reset
| ex : admin
| 2. NEW_PASSWORD is the password to change
| - password rule #1 : more than 8 length
| - password rule #2 : inclusion of one or more alphabet characters
| - password rule #3 : inclusion of one or more numerical digits
| - password rule #4 : inclusion of one or more special characters
+-----+
--
Input USER_ID for installation. (q:quit)
administrator
Input NEW_PASSWORD for installation. (q:quit)

The password has been changed successfully.
Execution is completed.!!s
```

7.5. Recommended OS Parameters for LENA Installation (CentOS)

For LENA installation, it is recommended to set the OS parameter max user processes to 8192 or higher.

Table 24. Recommended OS parameters (CentOS)

parameter	Recommended Value	Default
max user processes	8192	1024
open files	8192	1024

On CentOS, you can check the max user processes setting by running the command 'ulimit -a' as follows.

OS parameter. check max user processes (CentOS)

```
$ ulimit -a +
core file size          (blocks, -c) 0 +
data seg size           (kbytes, -d) unlimited +
scheduling priority     (-e) 0 +
file size                (blocks, -f) 8192 +
pending signals         (-i) 14891 +
max locked memory       (kbytes, -l) 64 +
max memory size         (kbytes, -m) unlimited +
open files              (-n) 1024 +
pipe size               (512 bytes, -p) 8 +
POSIX message queues    (bytes, -q) 819200 +
real-time priority      (-r) 0 +
stack size              (kbytes, -s) 10240 +
cpu time                (seconds, -t) unlimited +
max user processes      (-u) 1024 +
virtual memory          (kbytes, -v) unlimited +
file locks              (-x) unlimited
```

On CentOS, you can set the number of processes and open files with the commands 'ulimit -u' and 'ulimit -n'. To apply the changes permanently, add the ulimit commands to each user's profile (.profile, .bash_profile), or enforce them (CentOS).

OS parameter settings - number of processes and open files (CentOS)

```
$ cat $HOME/.bash_profile*
.. (omitted)*
ulimit -u 8192*
ulimit -n 8192*
```

As another method, open the /etc/security/limits.conf (CentOS) file and set the maximum number of processes (nproc) and the maximum number of open files (nofile).

OS parameter settings - number of processes and open files (CentOS)

```
$ cat /etc/security/limits.conf*
.. (omitted)*
*      soft nproc 8192*
*      hard nproc 8192*
*      soft nofile 8192*
*      hard nofile 8192*
```

7.6. LENA Files that Grow Periodically

Table 25. Files that grow periodically

Item	Path	Deletion Cycle	Estimated Monthly Growth	Remarks
Manager periodic inspection logging	LENA_HOME/repository/monitoringDB	N/A	10MB ~ 120MB	
Manager monitoring, diagnostic reports	LENA_HOME/repository/monitoringDB	7 days	N/A	Auto-deletion
Manager diagnostic statistics	LENA_HOME/repository/monitoringDB	Permanent	1MB or less	
Manager backup files	LENA_HOME/repository/backup/lena-manager	Permanent	300MB or less	
Manager Server templates	LENA_HOME/repository/container	Permanent	10MB / Service Cluster	Depends on number of Service Clusters
Manager log	LENA_HOME/logs/lena-manager	30 days	10MB ~ 100MB	
Agent log	LENA_HOME/logs/lena-agent	30 days	N/A	Auto-deletion
Installer log	LENA_HOME/logs/lena-installer	Permanent	1MB or less	
Server instance log	Server instance installation path LENA_HOME/servers/server_id/logs	Permanent	Depends on load	Path can be changed

7.7. WAS Image OS Reference

Other WAS solution images use the following operating systems.

Table 26. Status of other WAS solution images (as of 2020)

WAS Image	OS Image
jboss/wildfly	centos:7
open-liberty:full-java8-openj9	debian:buster (from adoptopenjdk/openjdk8-openj9)
store/oracle/weblogic:12.2.1.4	Oracle Linux
ibmcom/websphere-traditional	ubuntu:16.04
Tomcat base image e.g.) tomcat:9-jdk8	Base image tag uses the OpenJDK base tag FROM openjdk:8-jdk (FROM debian :buster)

Contact

- **Department in charge** : LG CNS System Solutions Business Team
- **Address** : LG Science Park E13, 10, Magokjungang 10-ro, Gangseo-gu, Seoul [07796]
- **Telephone** : (02) 2099-6136
- **Email** : lana-support@lgcns.com