




# WSO2 Product Administration

---

WSO2 Training

Module 06 - Deployment Patterns and Clustering

 CC BY 4.0

## What is a Cluster?

Multiple instances of WSO2 products can be installed in a cluster. It consists of multiple instances of a product that act as if they are a single instance and divide up the work.

- Better Performance
- Reliable
- High Availability
- Simplified Administration
- Increased scalability
- Failover and switchover

<https://docs.wso2.com/display/ADMIN44x/Clustering+Overview>



This approach improves performance, because the load is distributed within members of the cluster.

It also ensures reliability, because if one instance becomes unavailable or is experiencing high traffic, another instance will seamlessly handle the requests.

## Cluster Membership (1)

- To make a Carbon-based product instance a member of a cluster, it must be configured using one of these schemes:
  - **Well-Known Address** (WKA) membership scheme
  - **Multicast** membership scheme
  - **AWS** membership scheme
  - **Kubernetes** membership scheme
- WKA is a mechanism that allows cluster members to discover and join a cluster using unicast instead of multicast
  - Start the cluster with few members known as WKA members
  - Other members join the cluster through these WKA members
  - The system should have at least 2 WKA members

If the system has only 1 WKA member and this member goes down, the cluster breaks and the members will not be able to communicate with each other !

## Cluster Membership (2)

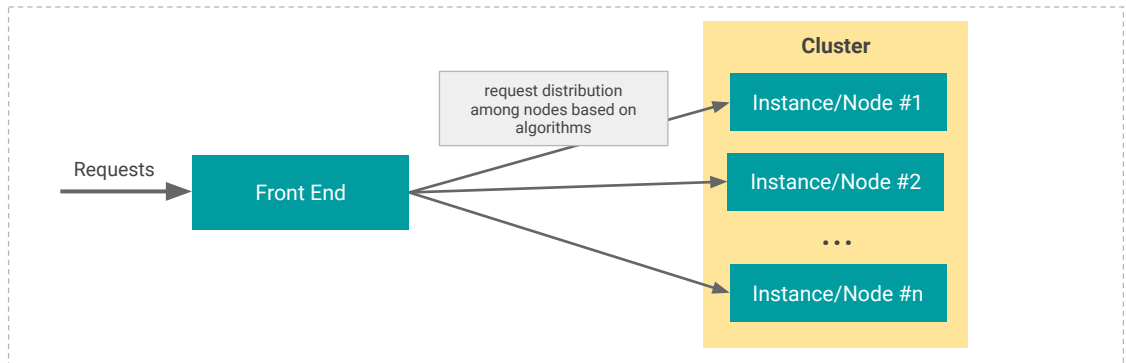
Multicast	WKA	AWS	Kubernetes
All nodes should be in the same subnet	Nodes can be in different networks	Amazon EC2 nodes	Kubernetes pods
All nodes should be in the same multicast domain	No multicasting requirement	No multicasting requirement	No multicasting requirement
Multicasting should not be blocked	No multicasting requirement	No multicasting requirement	No multicasting requirement
No fixed IP addresses or hosts required	At least one well-known IP address or host required	No fixed IP addresses or hosts required	No fixed IP addresses or hosts required

## Cluster Membership (3)

Multicast	WKA	AWS	Kubernetes
Failure of any member does not affect membership discovery	New members can join with some WKA nodes down, but not if all WKA nodes are down	Failure of any member does not affect membership discovery	Failure of any member does not affect membership discovery
Does not work on IaaS such as Amazon EC2	IaaS-friendly	Works on Amazon EC2	Works with Kubernetes and OpenShift Environments
No WKA requirement	Requires keepalive, elastic IPs, or some other mechanism for re-mapping IP addresses of WK members in cases of failure	No WKA requirement	No WKA requirement

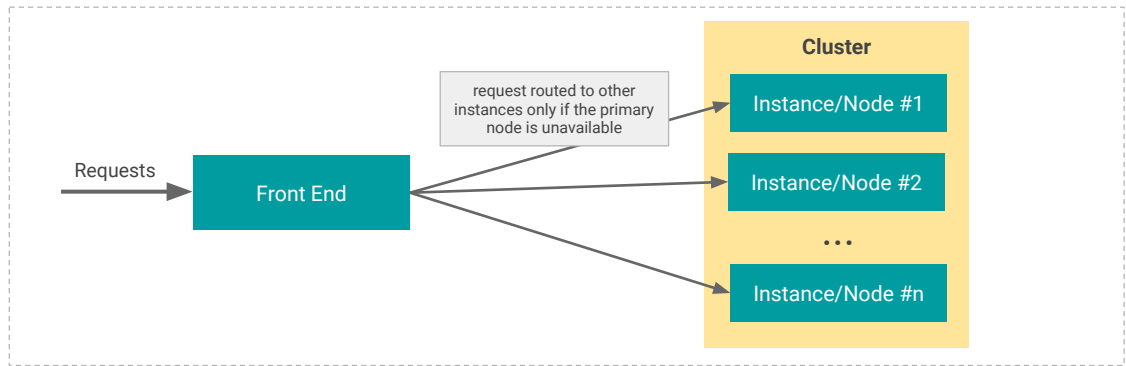
# Clustering

- For Load Balancing
  - The front end component distributes requests based on load distribution policies.
  - The front end is called Load Balancer

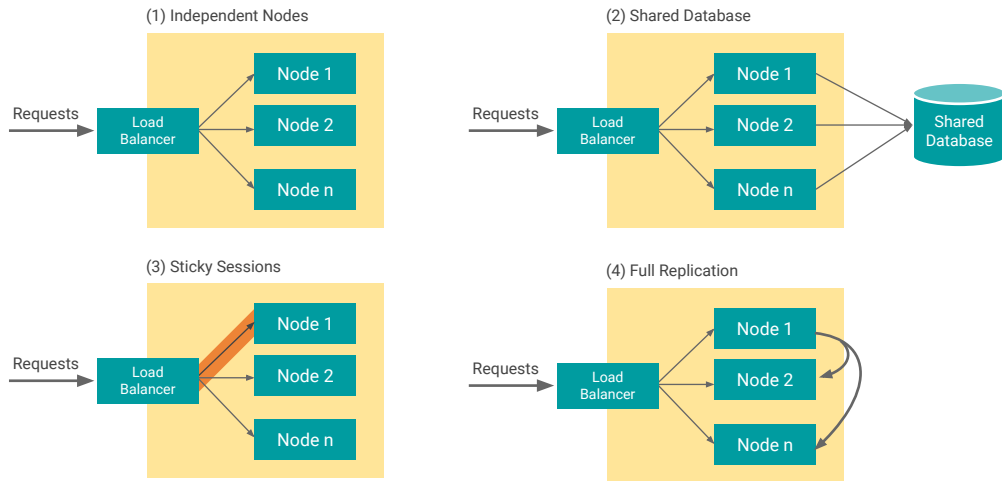


# Clustering

- For High Availability
  - User requests are redirected to a backup node if the primary node is unavailable



# Clustering Patterns

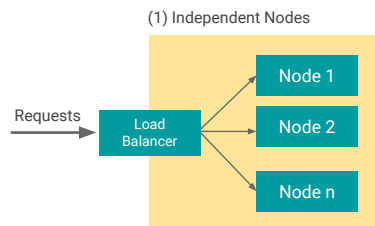




# Clustering

## (1) Independent nodes

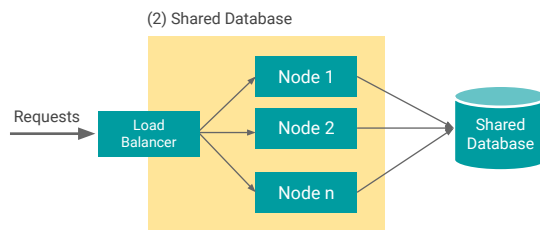
- No state is shared between the cluster nodes
- A Load Balancer is needed in front of the nodes, dispatching requests following given LB policies



# Clustering

## (2) Shared Database

- All the nodes of the cluster are connected to the same database
- Services hosted in the nodes are stateless
- The application state is stored in a database
- System can scale up till the database is overwhelmed

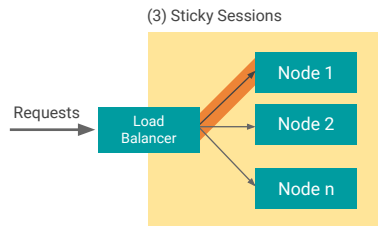


With enterprise class databases, this setup is known to scale to thousands on nodes, and this is one of the most common deployment setup. A common optimization is caching data from reads at the service level for performance, but unless nodes flush their caches immediately following a write, the reads may return old data from the cache. However, all the writes to the database must be transactional. Otherwise, concurrent writes might leave the database in a inconsistent state.

# Clustering

## (3) Sticky sessions

- Requests from the same client are tied to a given session
- This session is, in turn, tied to a given node, allowing requests to share the same state

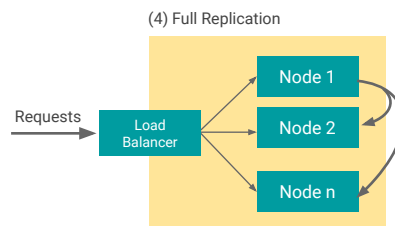


One downside of this approach is that if a node has failed, the sessions associated with that node are lost and need to be restarted. It is common to couple database based system described in scenario 2 with sticky sessions in practice, where session data is kept in memory, but persistent data are saved into a database.

# Clustering

## (4) Full Replication

- Changes done in a given node are replicated into the other nodes of the cluster
  - Implemented using a some kind of group communication methods.
- Group Communication keeps track of the cluster members. When a message is sent using group communication, it guarantees that all nodes in the current group will receive the message.



As seen in the previous slides, Hazelcast replaces Tribes as the distributed data management framework for WSO2 products based on Carbon 4.2.0

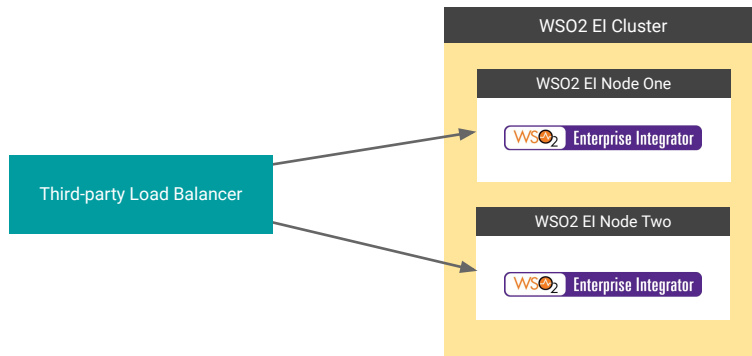
# API Manager Distributed Deployment

- **Step 1** - Install and configure WSO2 API-M
- **Step 2** - Install and configure the databases
- **Step 3** - Configure your deployment with production hardening
- **Step 4** - Create and import SSL certificates
- **Step 5** - Configure API-M Analytics
- **Step 6** - Configure the connections among the components and start the servers



<https://docs.wso2.com/display/AM260/Deploying+WSO2+API-M+in+a+Distributed+Setup>

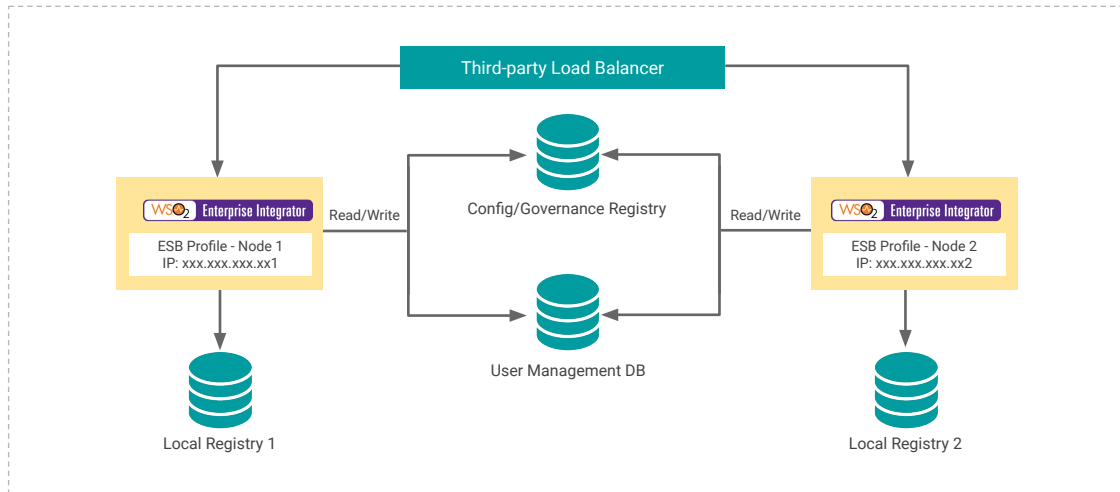
# Enterprise Integrator Distributed Deployment



You can set up a WSO2 Enterprise Integrator cluster with a third-party load balancer as depicted in the diagram.

<https://docs.wso2.com/display/EI650/Clustered+Deployment>

## Clustering the ESB Profile



## Clustering the ESB Profile

- **Step 1** - Configure the load balancer
- **Step 2** - Create the databases
- **Step 3** - Configure the ESB profile node
- **Step 4** - Deploy artifacts across the nodes
- **Step 5** - Test the cluster
- **Step 6** - Tune performance of the cluster

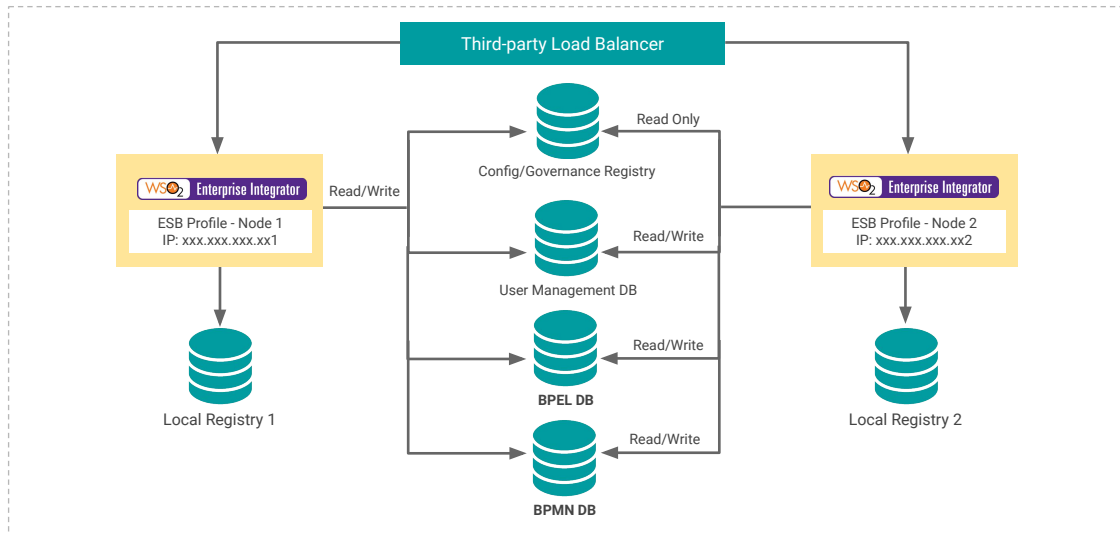


---

<https://docs.wso2.com/display/EI650/Clustering+the+ESB+Profile>



# Clustering the Business Process Profile



<https://docs.wso2.com/display/EI650/Clustering+the+Business+Process+Profile>  
e

# Clustering the Business Process Profile

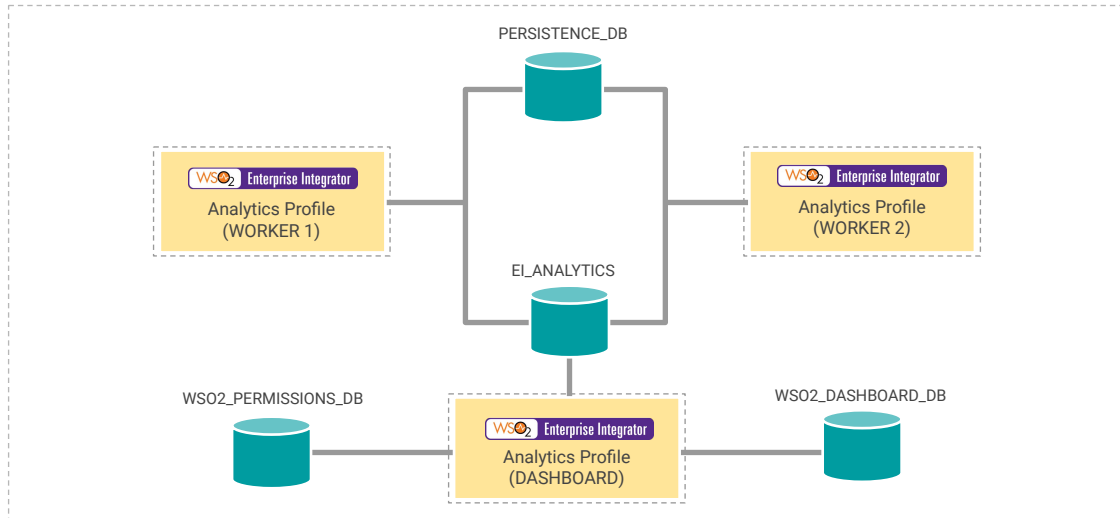
- **Step 1** - Configure the load balancer
- **Step 2** - Create the databases
- **Step 3** - Mount the registry
- **Step 4** - Configure the business process profile node
- **Step 5** - Configure Hazelcast properties
- **Step 6** - Deploy artifacts across the nodes
- **Step 7** -Test the cluster
- **Step 8** - Tune performance of the cluster



---

<https://docs.wso2.com/display/EI650/Clustering+the+Business+Process+Profile>

## Clustering the Analytics Profile



<https://docs.wso2.com/display/EI650/Clustering+the+Analytics+Profile>

# Clustering the Analytics Profile

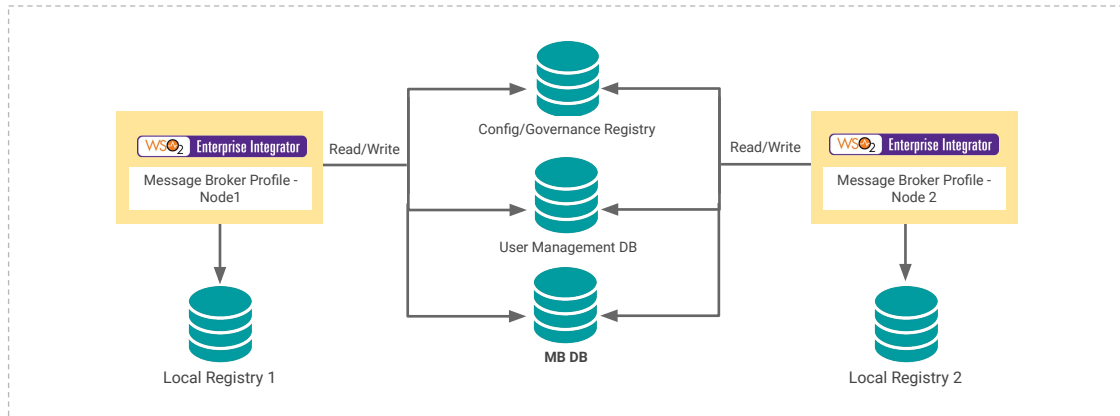
- **Step 1** - Set up the databases
- **Step 2** - Configure the Analytics dashboard
- **Step 3** - Configure the Analytics worker nodes
- **Step 4** - Start the cluster



---

<https://docs.wso2.com/display/EI650/Clustering+the+Analytics+Profile>

## Clustering the Message Broker Profile



## Clustering the Message Broker Profile

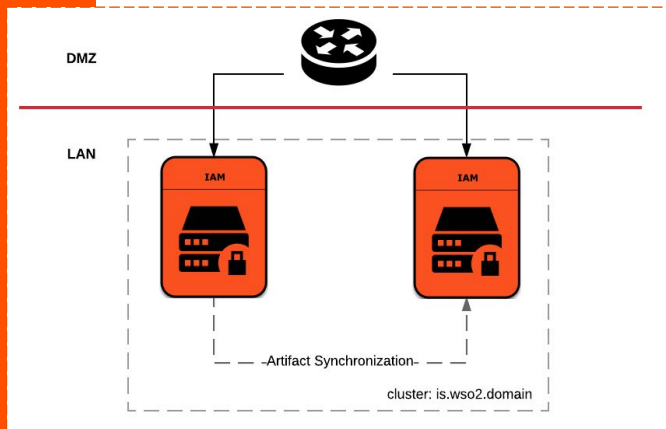
- **Step 1** - Create the databases
- **Step 2** - Mount the registry
- **Step 3** - Configure a message broker profile node
- **Step 4** - Testing the cluster



---

<https://docs.wso2.com/display/EI650/Clustering+the+Message+Broker+Profile>

# Clustering the Identity Server - Deployment Pattern 1



<https://docs.wso2.com/display/IS580/Deployment+Patterns>

# Clustering the Identity Server - Deployment Pattern 1

- **Step 1** - Configure the user store
- **Step 2** - Configure the datasources
- **Step 3** - Mount the registry
- **Step 4** - Cluster Identity Server for high availability
- **Step 5** - Change hostnames and ports
- **Step 6** - Enable artifact synchronization
- **Step 7** - Set up the dashboard
- **Step 8** - Front with a load balancer (Nginx)
- **Step 9** - Run the cluster
- **Step 10** - Start up and verifying product nodes

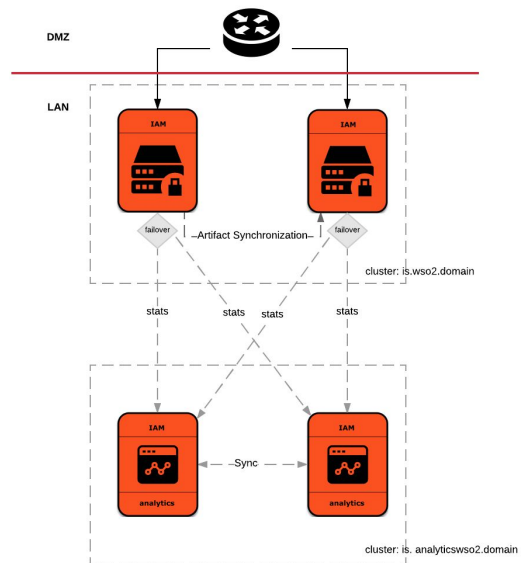


---

<https://docs.wso2.com/display/IS580/Setting+Up+Deployment+Pattern+1>



# Clustering the Identity Server - Deployment Pattern 2



<https://docs.wso2.com/display/IS580/Deployment+Patterns>

# Clustering the Identity Server - Deployment Pattern 2

- **Step 1** - Configure the user store
- **Step 2** - Configure the datasources
- **Step 3** - Mount the registry
- **Step 4** - Cluster Identity Server for high availability
- **Step 5** - Change hostnames and ports
- **Step 6** - Enable artifact synchronization
- **Step 7** - Set up the dashboard
- **Step 8** - Front with a load balancer (Nginx)
- **Step 9** - Run the cluster
- **Step 10** - Start up and verify product nodes
- **Step 11** - Configure minimum High availability deployment for WSO2 IS Analytics
- **Step 12** - Start the cluster
- **Step 13** - Test the HA deployment



---

<https://docs.wso2.com/display/IS580/Setting+Up+Deployment+Pattern+2>

# THANK YOU

wsO2.com

---



 CC BY 4.0