17

Managing Clusters

Objectives

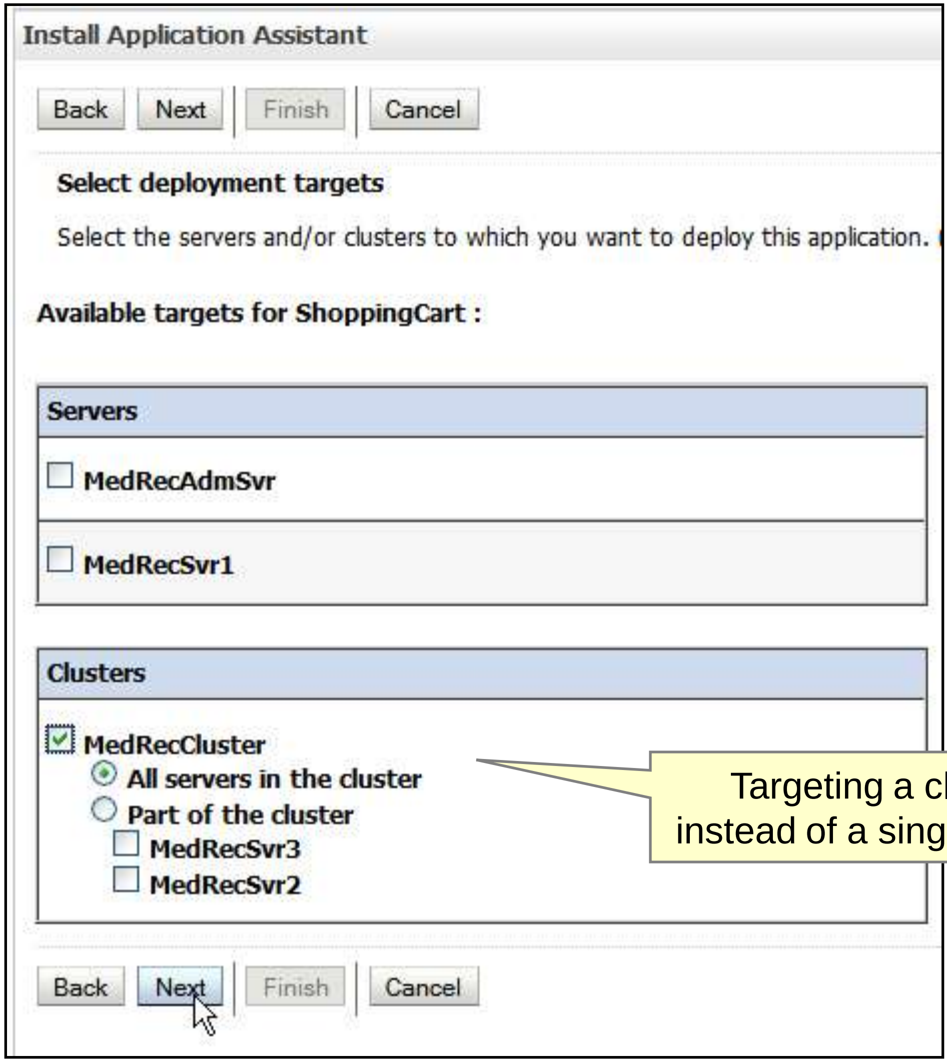
After completing this lesson, you should be able to do the following:

- Deploy applications to a cluster
- Describe the replication of a session state in a cluster
- Configure replication groups
- Configure in-memory replication
- Configure JDBC replication
- Configure file replication
- Configure a multitier cluster for EJB applications

Road Map

- Deploying applications
  - Selecting a cluster as the target
  - Two-phase deployment
  - Production redeployment
- HTTP session management
- EJB clustering
- Troubleshooting a cluster

Deploying Applications to a Cluster



Install Application Assistant

Select deployment targets

Select the servers and/or clusters to which you want to deploy this application.

Available targets for ShoppingCart :

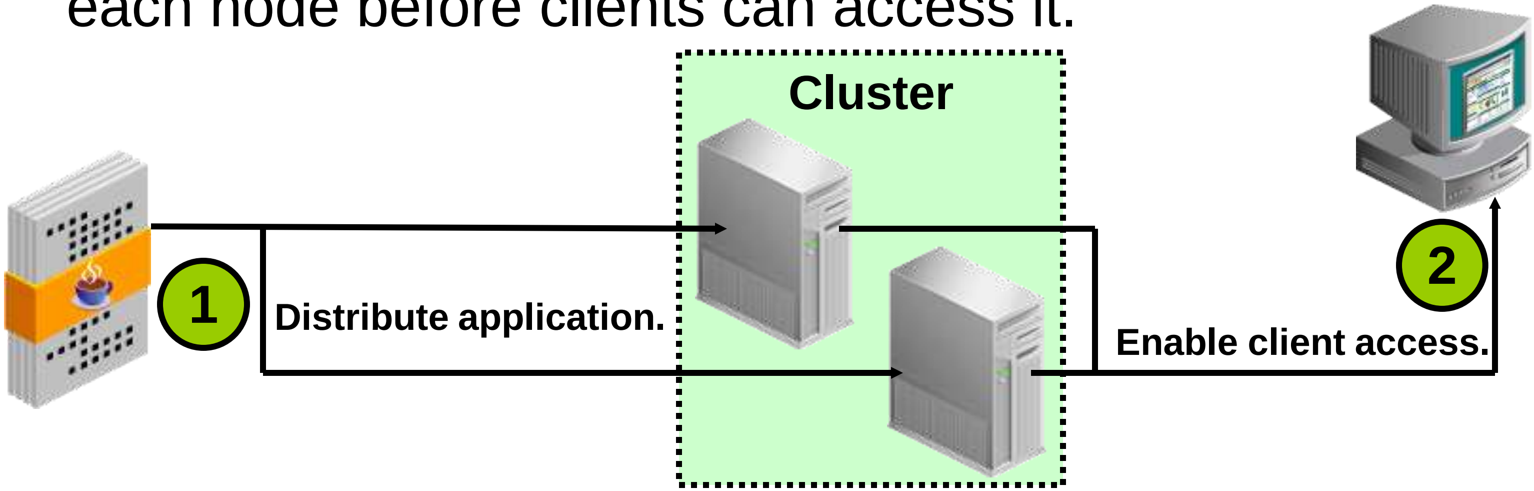
Servers
□ MedRecAdmSvr
□ MedRecSvr1

Clusters
☑ MedRecCluster ● All servers in the cluster ○ Part of the cluster □ MedRecSvr3 □ MedRecSvr2

Targeting a cluster instead of a single server

Two-Phase Deployment

- Applications are deployed using two-phase deployment (TPD).
  - Phase 1: Application components and modules are distributed to the server.
  - Phase 2: The application is deployed if phase 1 is successful and client access is permitted.
- This ensures that an application is available and active on each node before clients can access it.



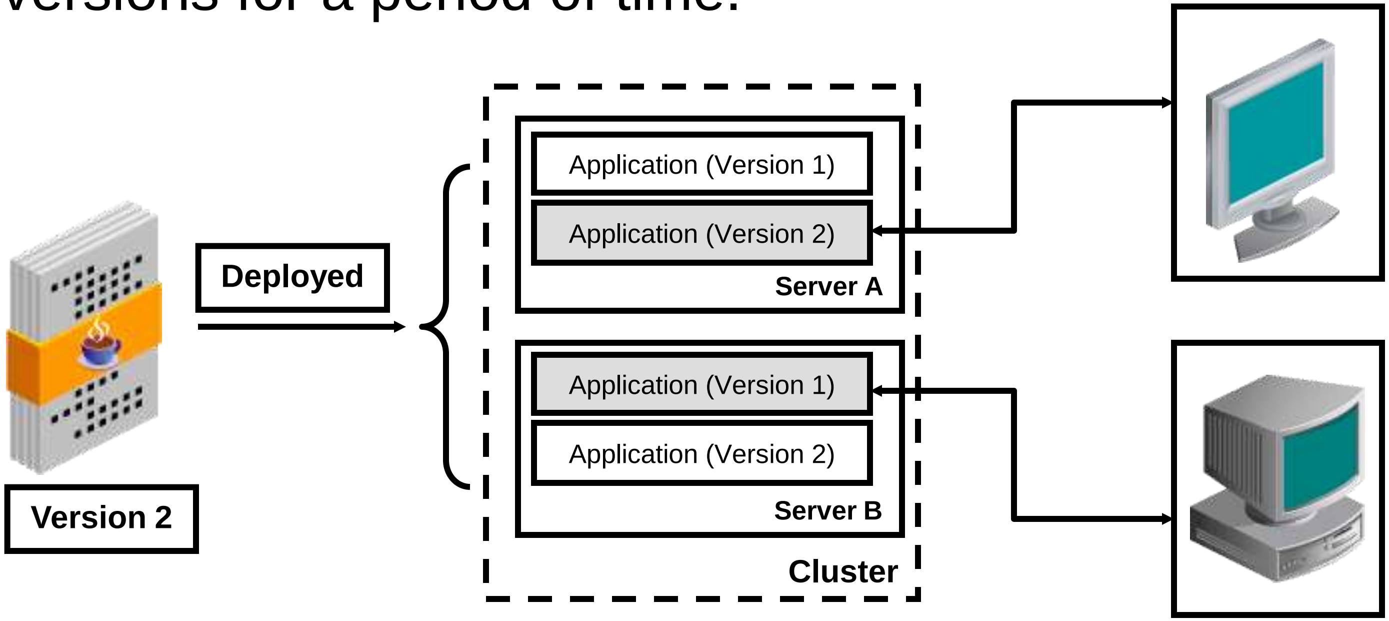
Considerations for Deploying to Cluster

- It would be good to have all the servers in the cluster running before an application is deployed to a cluster.
- If phase 2 fails on one server, the application is still deployed to other servers in the cluster.
- Do not change cluster membership while deploying applications to the cluster.
- Oracle WebLogic Server allows partial deployment of applications to a partitioned server by default.
- You can configure Oracle WebLogic Server to disallow partial deployments by using the `enforceClusterConstraints` tag.

Production Redeployment in a Cluster

When you use production redeployment of an application in a cluster, each server instance in the cluster retires the old version when the work is complete on that server.

- Therefore, different servers may be running different versions for a period of time.



Road Map

- Deploying applications to clusters
- HTTP session management
  - HTTP session failover
  - Replication groups
  - In-memory replication
  - Persistent replication
- EJB session replication
- Troubleshooting a cluster

HTTP Session Failover

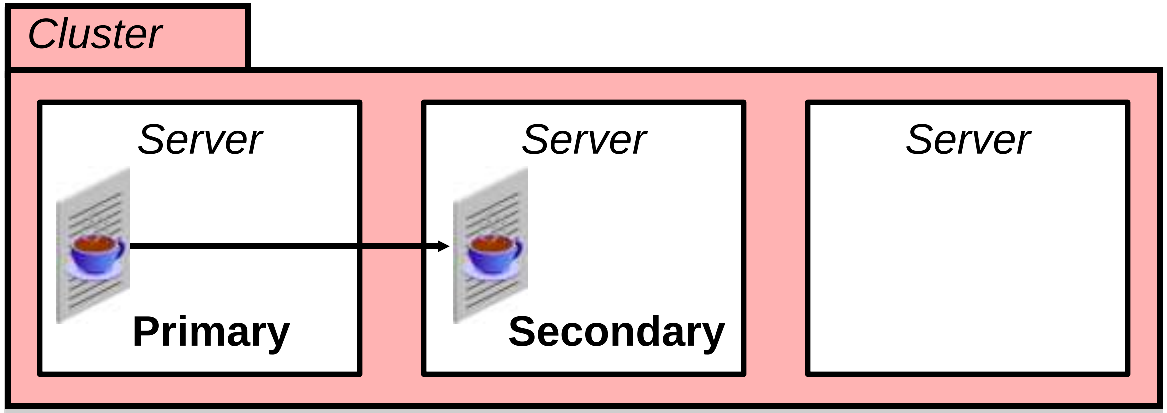
- Web applications use HTTP sessions to track information in server memory for each client.
- By default, when a client fails over to another server in the cluster, its session information is lost.
- Oracle WebLogic Server supports several *Session Replication* strategies to recover sessions from failed servers:
  - In-memory replication
  - JDBC replication
  - File replication
- Replication is configured for each Web application within its `weblogic.xml` file.

HTTP Session State Replication

- Session persistence is configured using the `<session-descriptor>` element in the `weblogic.xml` deployment descriptor file.
  - Each persistence method has its own set of configurable parameters.
- You should also configure access to the cluster through a collection of Web servers with identically configured proxy plug-ins or load-balancing hardware.
- Machine definition is one of the factors that WebLogic takes into account when it chooses another server as its backup for session information.

HTTP Session: In-Memory Replication

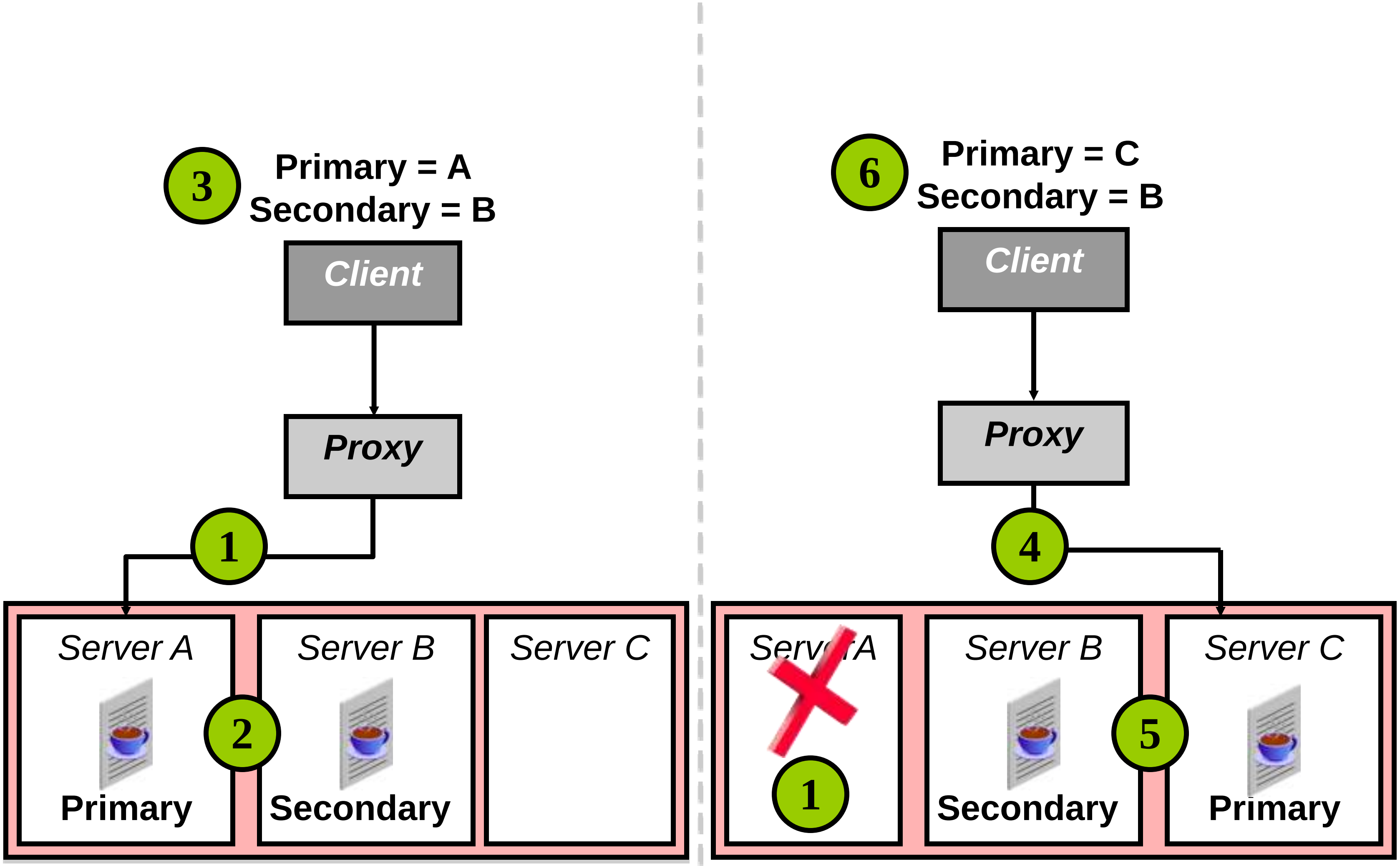
- Each user’s session always exists on two servers:
  - Primary
  - Secondary
- Every update to the primary session is automatically replicated on the secondary server, either synchronously (default) or asynchronously (batch).



In-Memory Replication and Proxy Servers

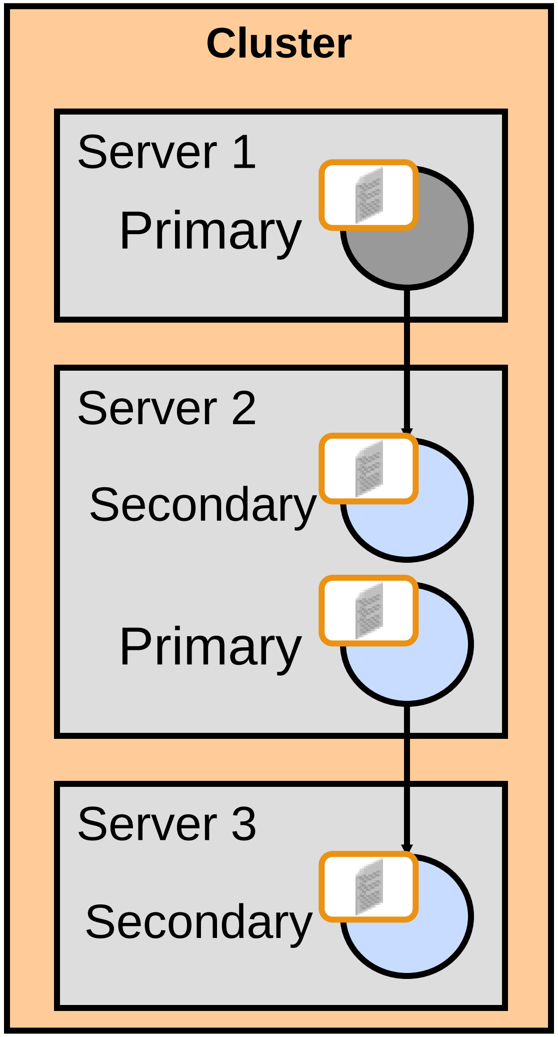
- Oracle WebLogic Server uses nonpersistent cookies to track the primary and secondary servers for each client.
- Subsequent requests from the same client must be directed to the same primary server by the proxy.
- The server that is being failed over to automatically assumes the role of the primary server.

In-Memory Replication: Example



In-Memory Replication

- WLS can replicate:
  - `HttpSession` objects
  - Stateful session EJBs
- Session objects exist on only two servers.
- Secondary:
  - The server is determined by the replication group and machine definition.
  - The object is created immediately after the primary object is created.
- Primary failure makes the backup object the primary object.



Requirements for In-Memory Replication

- Subsequent requests from the same client must have access to the same primary object.
- To use in-memory replication for the HTTP session state, clients must access the cluster using one of these:
  - The load-balancing hardware (WLS aware)
  - Oracle HTTP Server with the `mod_wl_ohs` module
  - A collection of Web servers, or a single Web server, with WebLogic proxy plug-ins (configured identically)
  - Oracle WebLogic Server configured with `HTTPClusterServlet`

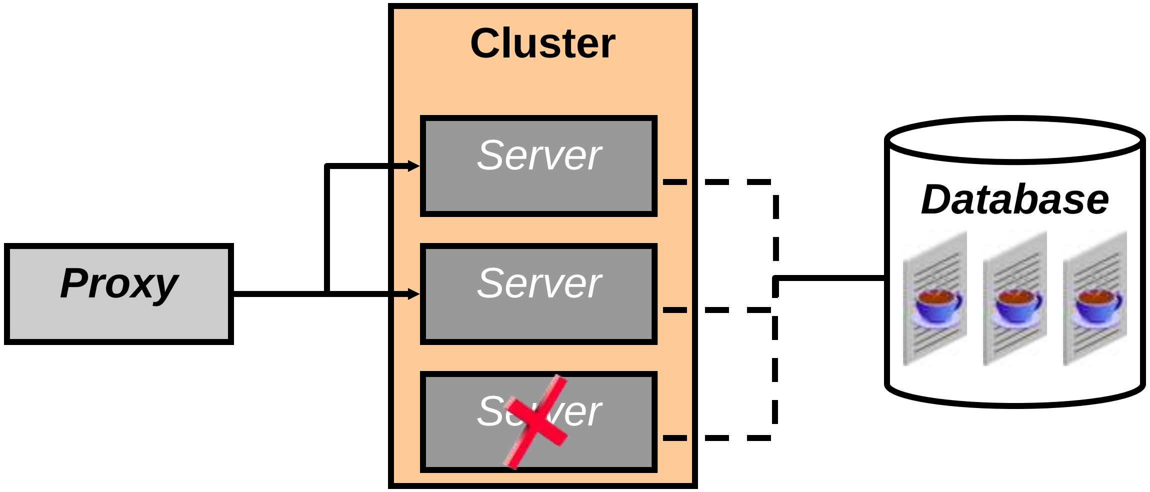
Configuring In-Memory Replication

1. Configure the proxy server (if applicable).
2. Optionally, define replication groups or machines, or both.
3. Specify the persistence type in the `weblogic.xml` deployment descriptor; the options include:
   - `replicated`
   - `replicated-if-clustered`
   - `async-replicated`
   - `async-replicated-if-clustered`

```
...
<session-descriptor>
   <persistent-store-type>replicated</persistent-store-type>
</session-descriptor>
...
```

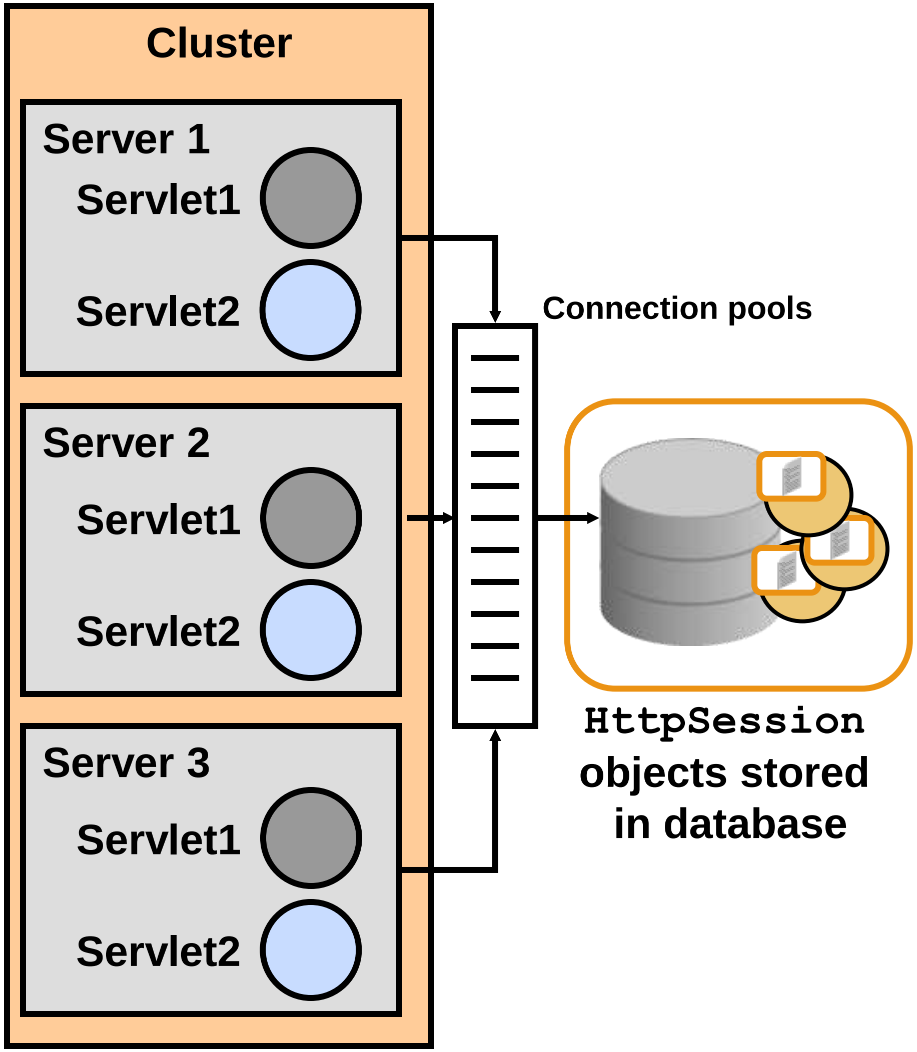
HTTP Session: Replication Using JDBC

- HTTP sessions can be persisted to a database using a common JDBC data source.
- The required Data Definition Language (DDL) file is available in the documentation.
- *All* members of the cluster have access to any client's session for failover purposes (no primary or secondary).



HTTP Session Replication Using JDBC

- All server instances have access to all sessions.
- Subsequent requests from the same client can be handled by any server.
  - Great failover capability
  - Significant performance reduction
- Changing session objects causes (slow) database synchronization.



Configuring JDBC Replication

1. Create the required table in the database.
2. Create a JDBC data source that has read/write privileges for your database.
3. Configure JDBC session persistence in the `weblogic.xml` deployment descriptor.

```
...
<session-descriptor>
   <persistent-store-type>jdbc</persistent-store-type>
   <persistent-store-pool>MyDataSource</persistent-store-pool>
</session-descriptor>
...
```

JDBC Persistent Table Configuration

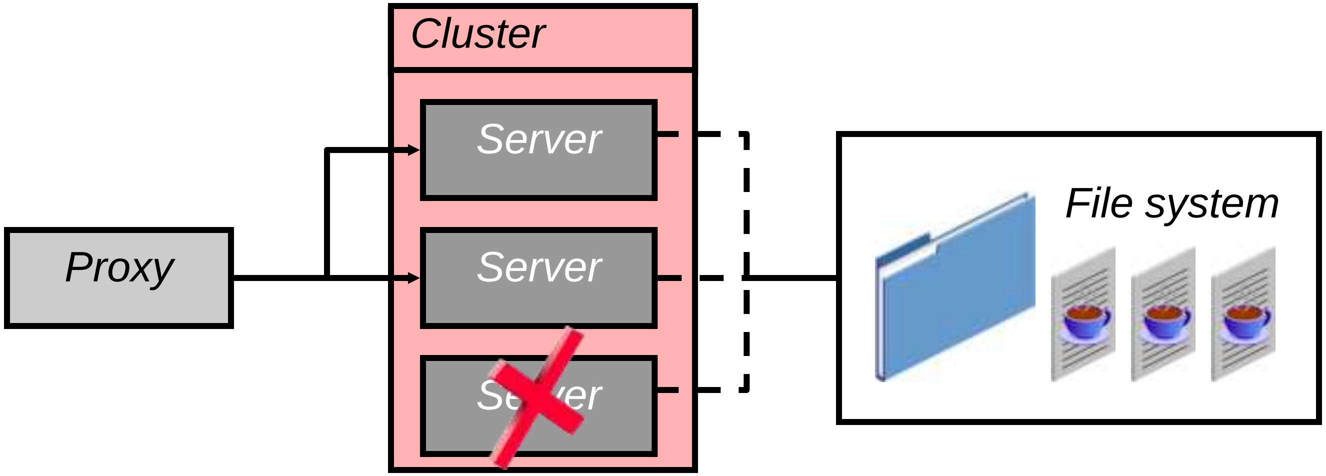
A database table named WL_SERVLET_SESSIONS must exist with read/write access:

Column Head	Column Data Type
WL_ID	char, 100 variable width char
WL_CONTEXT_PATH	
WL_CREATE_TIME	numeric, 20 digits
WL_IS_VALID	char, 1 character
WL_SESSION_VALUES	BLOB, very large
WL_ACCESS_TIME	numeric, 20 digits
WL_IS_NEW	numeric, 20 digits

Primary key

HTTP Session Replication Using File

File replication is similar to JDBC replication, but it persists sessions to a highly available file system.



Configuring File Replication

1. Create a folder shared by all servers on the cluster on a highly available file system.
2. Assign read/write privileges to the folder.
3. Configure file session persistence in the `weblogic.xml` deployment descriptor.

```
...
<session-descriptor>
   <persistent-store-type>file</persistent-store-type>
   <persistent-store-dir>/mnt/wls_share</persistent-store-dir>
</session-descriptor>
...
```

Replication Groups

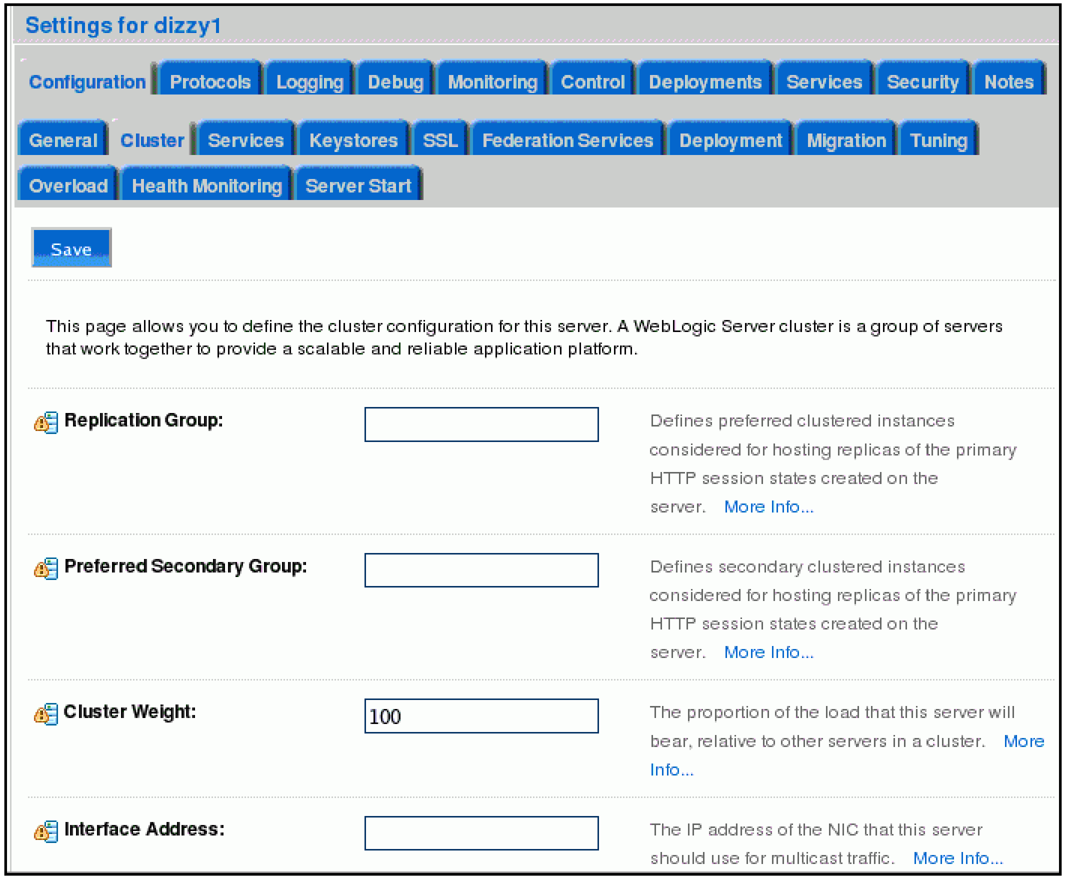
- A replication group is a logical grouping of related servers in a cluster.
- WLS enables you to determine where to put backup objects using replication groups.
- WLS attempts to:
  - Send backup objects to a preferred secondary replication group, if it is configured
  - Send backup objects to a different machine
  - Avoid sending backup objects to servers in the same replication group

Replication Groups

- Replication groups:
  - Represent a subset of servers within a cluster
  - Help to determine the placement of secondary sessions (for example, avoid replicating within the same room)
  - Are not explicitly defined in the console-like machines and clusters
- WLS attempts to:
  - Send secondary sessions to servers that are assigned to the *preferred secondary replication group* of the primary server
  - Avoid sending secondary sessions to servers that are assigned to the same replication group as the primary server

Configuring Replication Groups

Select each server in a cluster and assign each a pair of replication groups.

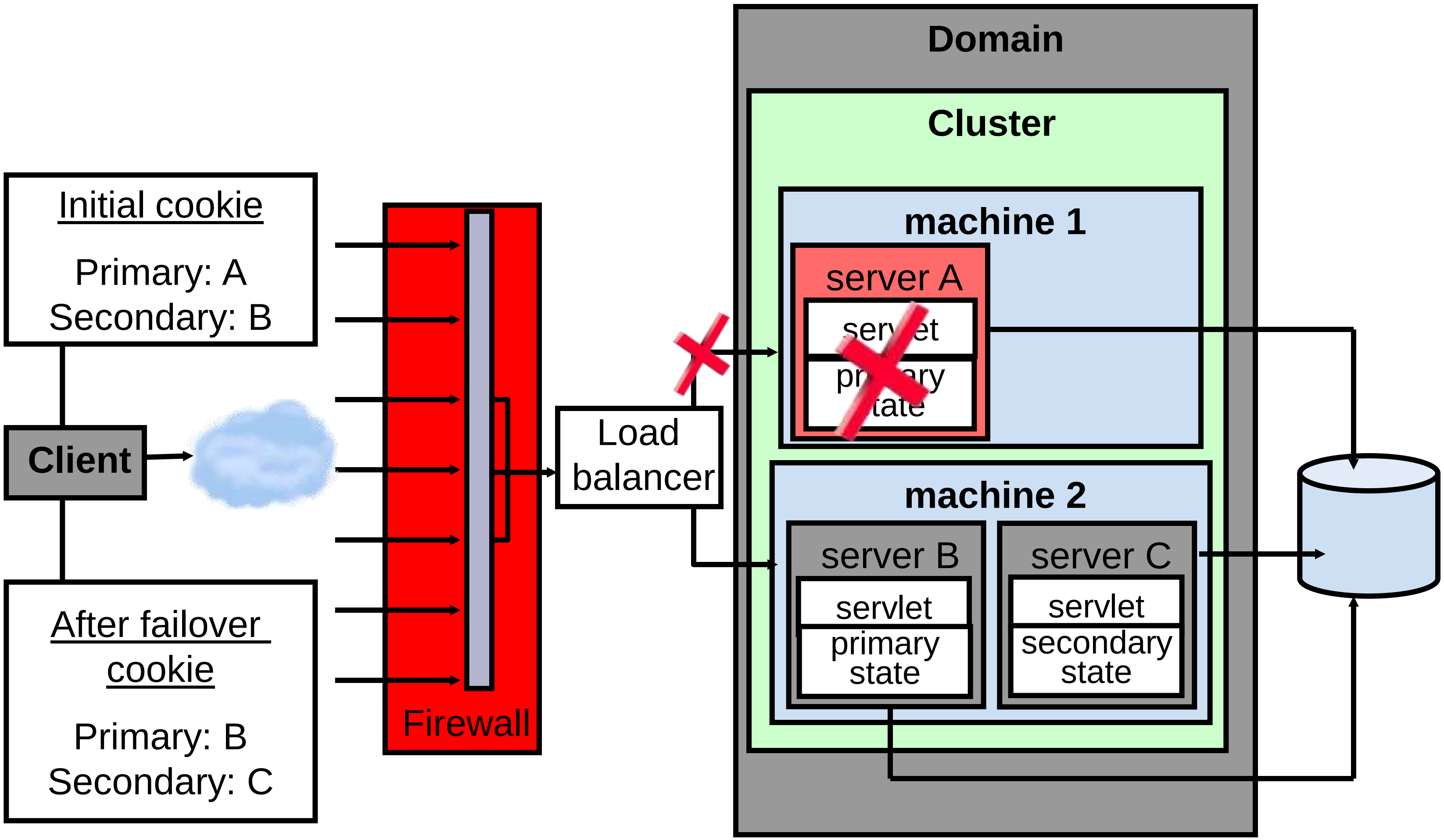


Settings for dizzy1

This page allows you to define the cluster configuration for this server. A WebLogic Server cluster is a group of servers that work together to provide a scalable and reliable application platform.

Replication Group:		Defines preferred clustered instances considered for hosting replicas of the primary HTTP session states created on the server. More Info...
Preferred Secondary Group:		Defines secondary clustered instances considered for hosting replicas of the primary HTTP session states created on the server. More Info...
Cluster Weight:	100	The proportion of the load that this server will bear, relative to other servers in a cluster. More Info...
Interface Address:		The IP address of the NIC that this server should use for multicast traffic. More Info...

Failover with Replication Groups



HTTP State Management Best Practices

- Create WLS machines if you are replicating the state across servers on different physical machines.
- Use replication groups to define the failover strategy.
- Choose the most appropriate replication strategy depending on the application needs and architecture.
- Use the `ServerDebugConfig` MBean to track session replication problems.
- Ensure that objects placed in replicated sessions are serializable.

Road Map

- Deploying applications to clusters
- HTTP session management
- EJB session replication
  - EJB clustering deployment descriptors
  - Configuring stateless session beans
  - Configuring stateful session beans
- Troubleshooting a cluster

Configuring EJB Clustering in Deployment Descriptors

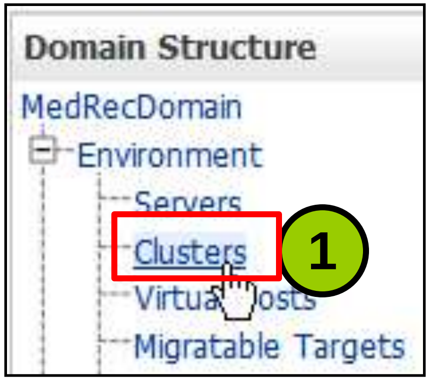
- Clustering of EJB based on version 2 are configured in the application-specific deployment descriptors.
- When using clustering based on EJB version 3.0, you can use the deployment plans to implement clustering.

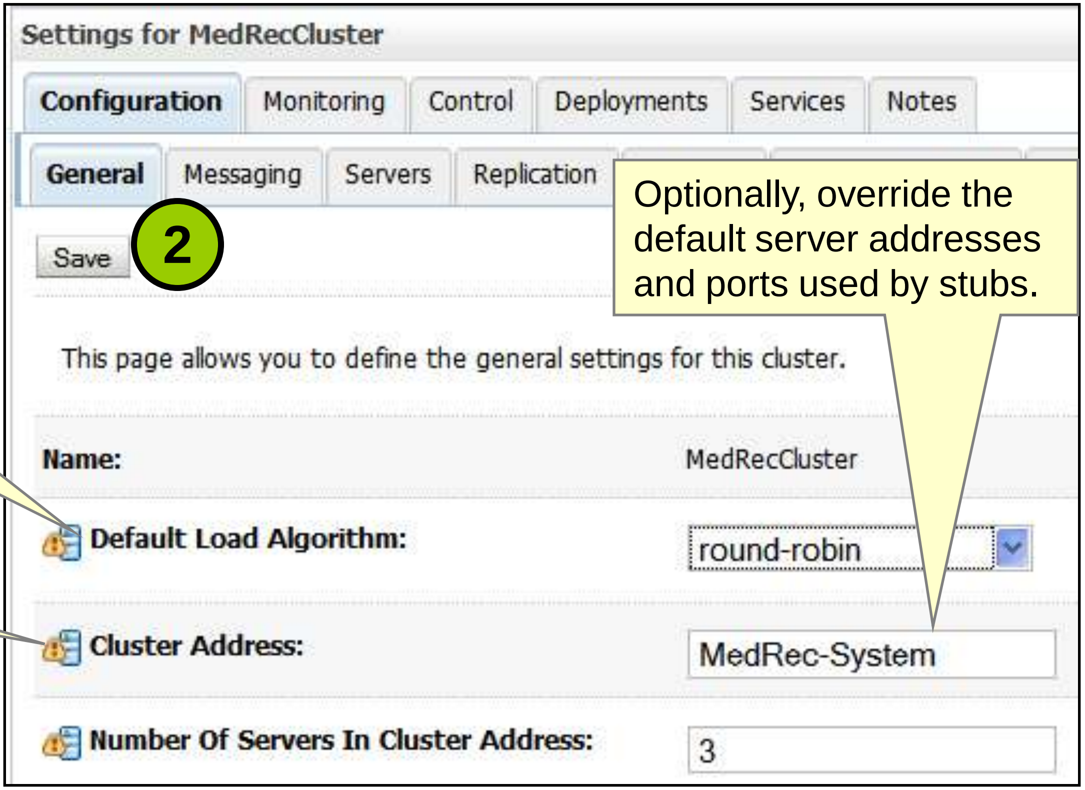
A snippet from `weblogic-ejb-jar.xml`:

```
...
<stateless-clustering>
   <stateless-bean-is-clusterable>True
   </stateless-bean-is-clusterable>

   <stateless-bean-load-algorithm>random
   </stateless-bean-load-algorithm>
   ...
</stateless-clustering>
...
```

Configuring EJB Clustering Using the Administration Console





This page allows you to define the general settings for this cluster.

Name: MedRecCluster

Default Load Algorithm: round-robin

Cluster Address: MedRec-System

Number Of Servers In Cluster Address: 3

Optionally, override the default server addresses and ports used by stubs.

Default for all EJBs, if not overridden

Required only if **Address** is a single DNS name

Configuring Clusterable Stateless Session EJBs

- The WLS-specific deployment descriptor has a tag for configuring stateless session EJB clustering parameters.
- A snippet from a typical `weblogic-ejb-jar.xml` file:

```
…
<stateless-session-descriptor>
<!-- Other Tags As Appropriate Here… -->
  <stateless-clustering>
    <stateless-bean-is-clusterable>True</stateless-bean-is-
    clusterable>
    <stateless-bean-load-algorithm>random</stateless-bean-load-
    algorithm>
    <stateless-bean-call-router-class-
    name>beanRouter</stateless-bean-call-router-class-name>
    <stateless-bean-methods-are-idempotent>True</stateless-
    bean-methods-are-idempotent>
  </stateless-clustering> ...
```

Clusterable EJBs: Idempotent Methods

Example of an idempotent method snippet in `weblogic-ejb-jar.xml`:

```
…
<!-- LAST TAG inside <weblogic-ejb-jar.xml> -->
  <idempotent-methods>
    <method> <!-- can be repeated -->
      <ejb-name>exampleSession</ejb-name>
      <method-intf>Remote</method-intf>
      <method-name>processUser</method-name>
      <method-params>
        <method-param>java.lang.String</method-param>
      </method-params>
    </method>
  </idempotent-methods>
…
</weblogic-ejb-jar>
```

Stateful Session Beans

- Each stateful session EJB is unique.
- All calls on a remote stub must be directed to the server that contains the EJB.

A stateful session EJB is "pinned" to the server that it is created on. Its remote stub must also be pinned to the same server.

Configuring Clusterable Stateful Session EJBs

- The WLS-specific deployment descriptor has a tag for configuring stateful session EJB clustering parameters.
- The replication type for EJBs is InMemory or None.

```
<stateful-session-descriptor>
  <stateful-session-clustering>
    <home-is-clusterable> true    </home-is-clusterable>
    <home-load-algorithm> random  </home-load-algorithm>
    <home-call-router-class-name> common.QARouter
    </home-call-router-class-name>
    <replication-type>    InMemory </replication-type>
  </stateful-session-clustering>
</stateful-session-descriptor>
```

Read/Write Versus Read-Only

- There are two types of entity beans to consider:
  - Read/write
  - Read-only
- For read/write entity beans, load balancing and failover occur only at the home level.
- For read-only entity beans, the replica-aware stub:
  - Load balances on every call
  - Does not automatically fail over in the event of a recoverable call failure

Entity Bean Cluster-Aware Home Stubs

- Entity beans can have cluster-aware home stubs that have knowledge of the EJB Home objects on all WLS instances in the cluster.
- The `home-is-clusterable` deployment element in the `weblogic-ejb-jar.xml` file determines whether a home stub is cluster-aware.
- An example of setting an entity EJB home stub as cluster-aware:

```
<entity-clustering>
   <home-is-clusterable>True</home-is-clusterable>
   <home-load-algorithm>random</home-load-algorithm>
   <home-call-router-class-name>beanRouter
   </home-call-router-class-name>
</entity-clustering>
```

EJB Best Practices

- Set pool and cache sizes in accordance with anticipated load and execute threads per server.
- Understand that cache sizes equally affect all nodes in the cluster.
- Mark bean methods that can be called multiple times with impunity as idempotent in their deployment descriptors.

Quiz

Select all valid values for the persistent store type element in `weblogic.xml`.

1. `file`
2. `replicated`
3. `unicast`
4. `async-replicated-if-clustered`
5. `jdbc`
6. `async-wan`

Quiz

Which two Oracle WebLogic Server features can be used to control the destination servers that are used for in-memory replication?

1. Web service
2. Replication group
3. Data source
4. Node Manager
5. Machine

Quiz

Which of the following terms is NOT associated with in-memory replication?

1. Cookie
2. Secondary
3. Session
4. Schema
5. Primary
6. Synchronous

Quiz

Which types of replication configuration are allowed for EJBs?

1. JDBC
2. File
3. InMemory
4. None

Summary

In this lesson, you should have learned how to:

- Deploy applications to a cluster
- Describe session state in a cluster
- Configure replication groups
- Configure in-memory replication
- Configure JDBC replication
- Configure file replication

Practice 17: Overview
Managing Clusters

This practice covers the following topics:

- Defining a cluster as a target for new applications
- Retargeting existing applications to a cluster
- Deploying an application to a cluster
- Setting up in-memory session replication