Module 15

Clustering Services

At the end of this module, you will be able to:

- ✓ Set up whole server migration
- ✓ Understand JNDI clustering
- ✓ Understand JDBC clustering
- ✓ Migrate a JMS server
- ✓ Migrate transactions

Road Map

1. **Clustering Services**
   - Clusterable Services
   - Service Level Migration
   - Server Level Migration
2. JNDI
3. JDBC
4. Transactions
5. JMS

Which Services Can Be Clustered?

- A *clustered service* is an API or interface that is available on multiple servers in a cluster.
- WebLogic Server provides clustering services for:
  - Web applications
  - EJB and RMI objects
  - JNDI tree
- WebLogic Server provides limited clustering services for:
  - JDBC connections
  - JMS connection factories

Service Failover

- Services that are deployed to all servers in a cluster automatically have transparent failover.
- JMS and JTA cannot be deployed to all servers in a cluster
  - They are pinned services
- There are two ways to handle JMS and JTA failover:
  - Service Level Migration
  - Server Level Migration

Service Level Migration

- Service-level migration is performed manually.
- Pinned services are moved from the failed server to a new server.
  - The new server then takes over and processes any data left over when the server failed then takes over any future processing
- A migratable service is accessed by clients using a migration-aware RMI stub.

Server Level Migration

- Server level migration moves the whole clustered server instance, in it’s entirety.
  - It is an alternative to service level migration
  - They should not be used together
- Server level migration is:
  - Only supported using the Secure Shell (SSH) version of Node Manager
  - Not supported on Windows
- Server level migration can happen both manually and automatically.
- When a migratable server fails:
  - The server is automatically restarted on the same machine
  - If it cannot be started on the same machine, it moves to another machine

Server Level Migration Services and Resources

- Migratable Servers
  - Managed servers that host pinned services
- Cluster Master
  - One server instance that orchestrates the process of automatic server migration
- Target Machines
  - Allowable or preferred hosts for migratable servers
- Node Manager
  - A process that starts and stops server instances
- Lease Table
  - A database table where migratable servers persist their state information
- Administration Server
  - Configures migratable servers and target machines as well as obtains the state of migratable servers and orchestrates the manual migration process

Server Migration Environment

- Before configuring server migration, your environment must be set up properly.
  - Each managed server must use the same subnet mask for multicast communications
  - Each server hosting migratable services should be time synched
  - All operating system versions must support identical functionality for `ifconfig`
  - Interface names for each server should be the same
  - The database you are using to support migration must be supported

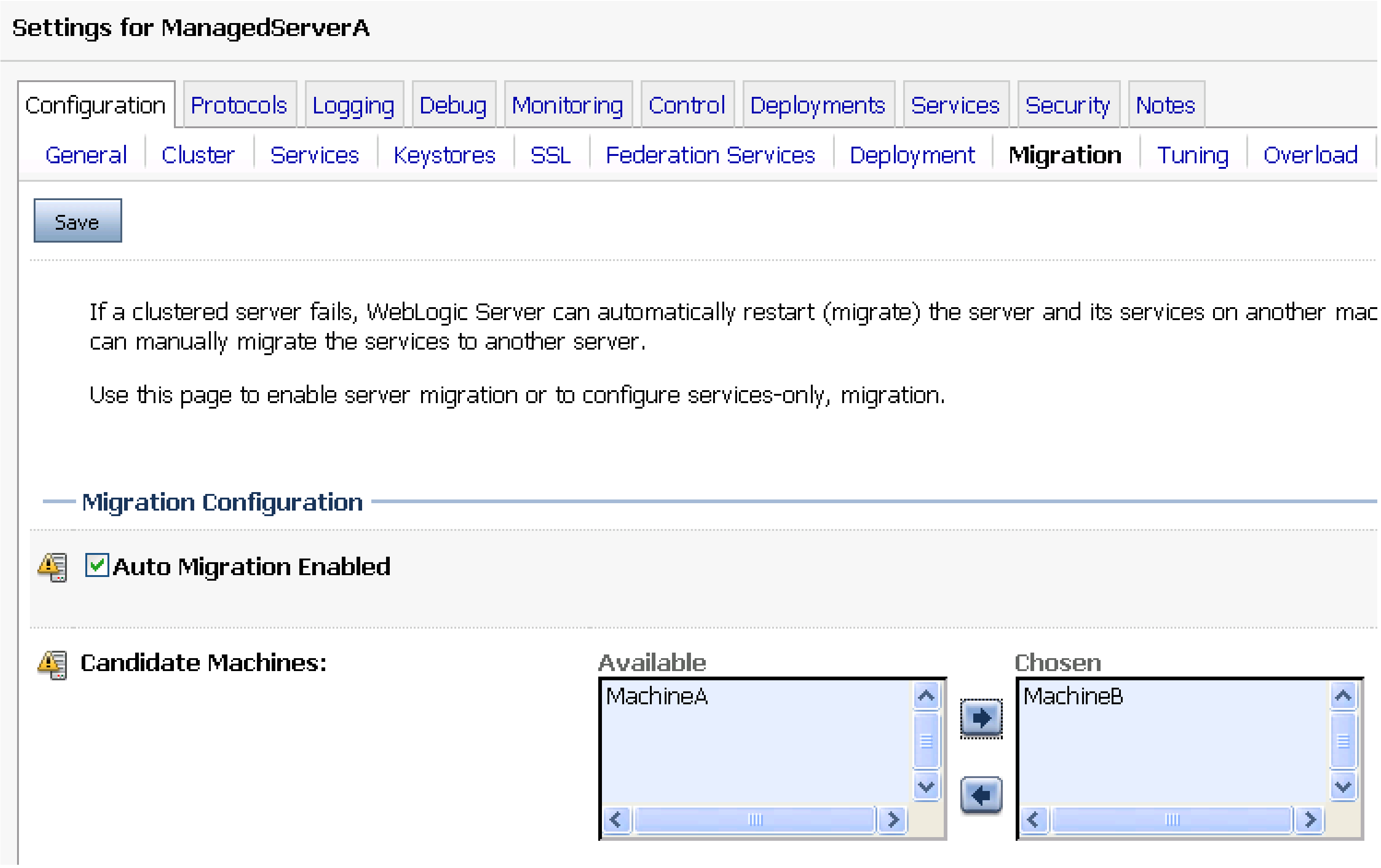
Server Migration Configuration…

- To configure server migration:
  - Configure each machine using server migration with a floating IP address
  - Configure the SSH version of node manager on each managed server (it must be running as well)
  - Configure the database by creating the leasing tables used to store machine-server associations
  - Configure a data source to point to the server migration database
  - Grant superuser privileges to the `wlsifconfig.sh` script
  - Ensure that `wlsifconfig.sh, wlscontrol.sh and nodemanager.domains` are included in your machine's PATH

…Server Migration Configuration…

- To configure server migration (continued):
  - Ensure that the machines hosting migratable services trust each other
  - Set the candidate machines for server migration.
  - Edit `wlscontrol.sh` and set the Interface variable to the name of your network interface
  - Configure the server properties to enable server migration

…Server Migration Configuration



Settings for ManagedServerA

If a clustered server fails, WebLogic Server can automatically restart (migrate) the server and its services on another mac can manually migrate the services to another server.

Use this page to enable server migration or to configure services-only, migration.

Migration Configuration

☑ Auto Migration Enabled

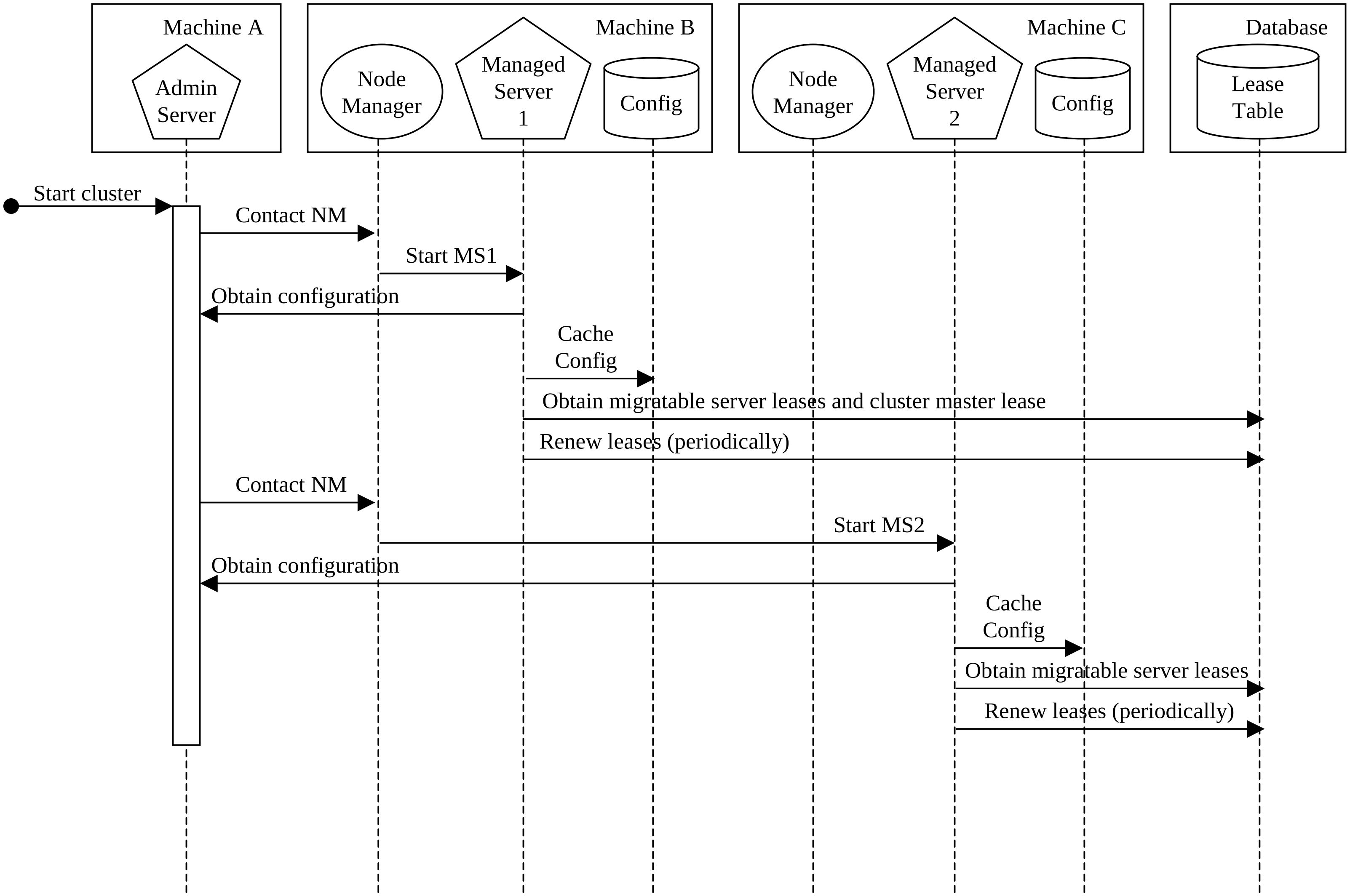
Candidate Machines:

Available	Chosen
MachineA	MachineB

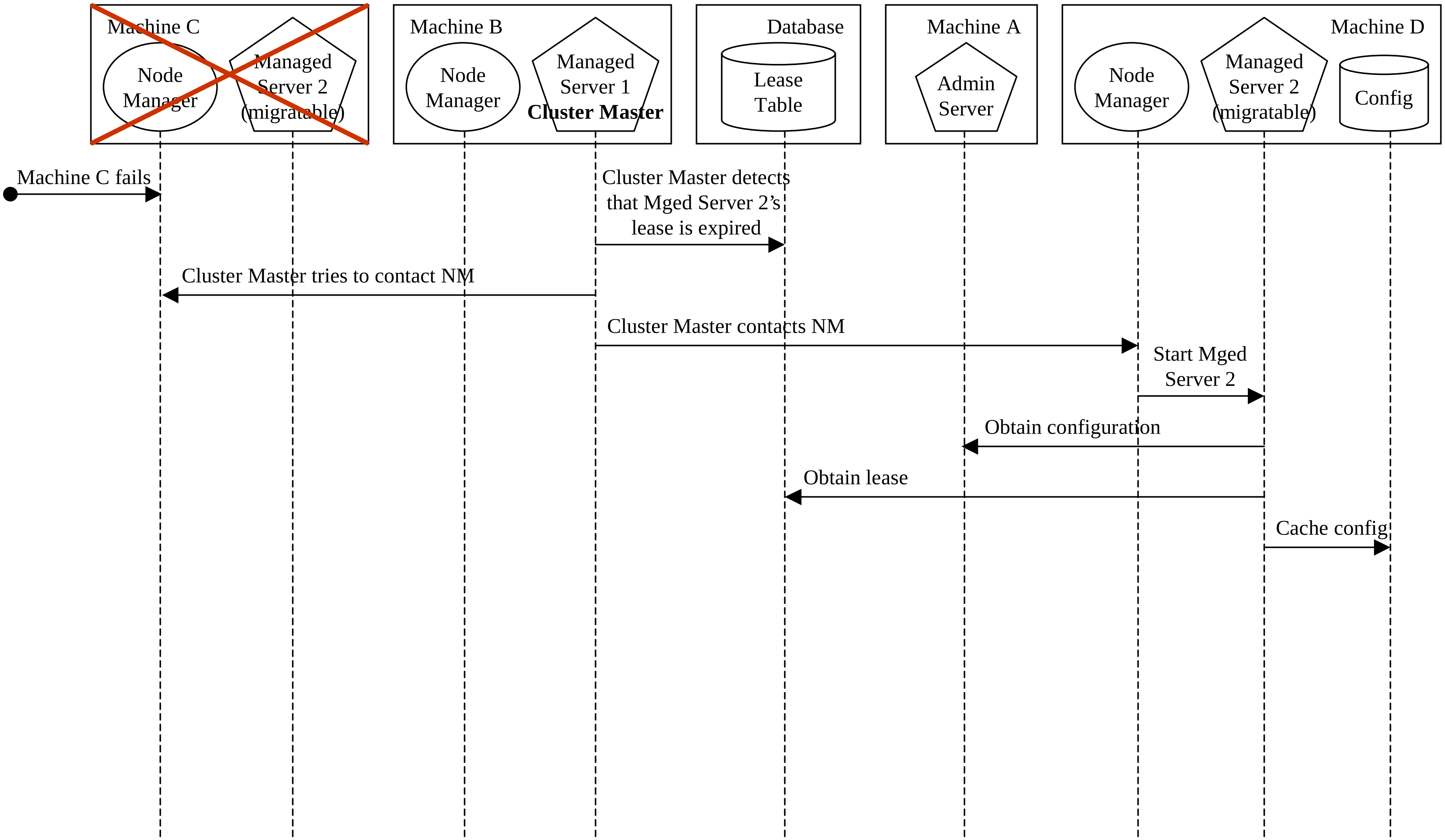
Server Migration Restrictions

- Each managed server must be configured to use the same subnet mask.
- Channels/Network Access Points must not have different Listen Addresses on a migratable server
- There is no built-in mechanism for transferring files
  - It is recommend that you store files on a different disk that is accessible by all the machines
- Node Manager security files must be copied to each machine
  - You can use the `nmEnroll()` WLST command

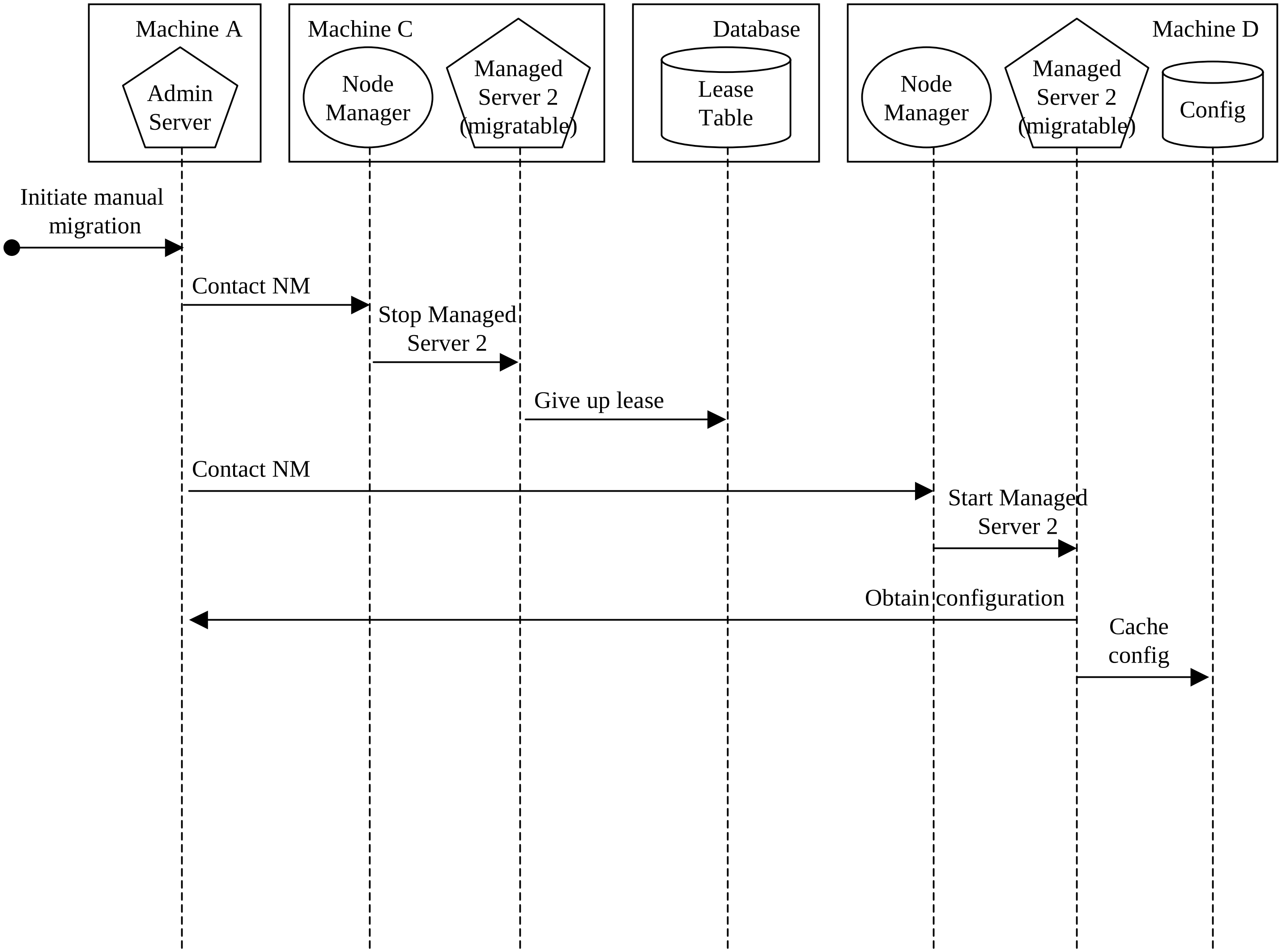
Migratable Server Startup Process



Automatic Server Migration



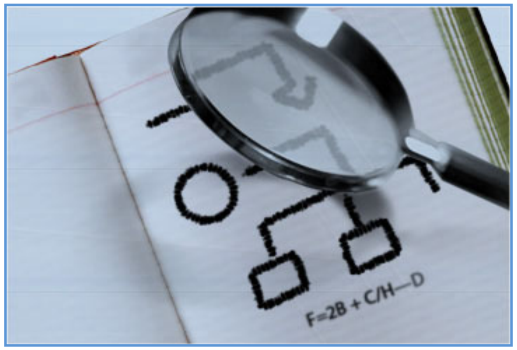
Manual Server Migration



Section Review

In this section, we learned how to:

- ✓ Understand what services are clusterable
- ✓ Understand the difference between service level migration and server level migration
- ✓ Set up server level migration



Road Map

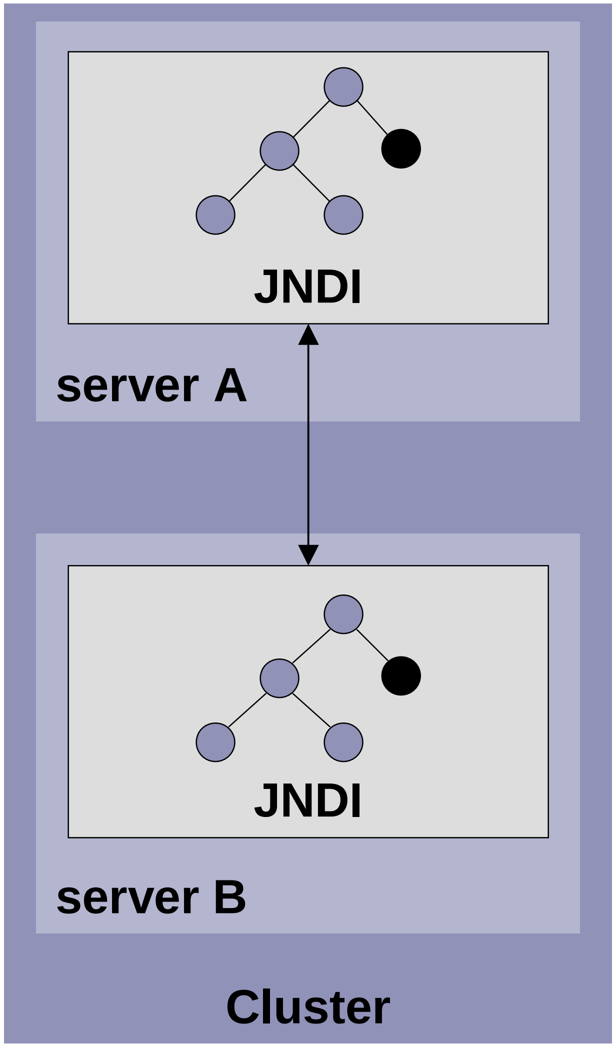
1. Clustering Services
2. **JNDI**
   - Cluster-Wide JNDI Server
   - JNDI Naming Conflicts
3. JDBC
4. Transactions
5. JMS

JNDI Clustering Support

- The JNDI tree contains factory objects for accessing these items:
  - JDBC DataSources
  - EJBs, RMI objects
  - JMS connection factories
- WebLogic Server replicates clusterable objects to all servers on a cluster, transparently.

Cluster-Wide JNDI Service

- Each WLS server issues messages that announce these events:
  - A new object bound into the naming tree
  - An object removed from the naming tree
  - An object rebound (updated) into the naming tree



JNDI Naming Conflicts

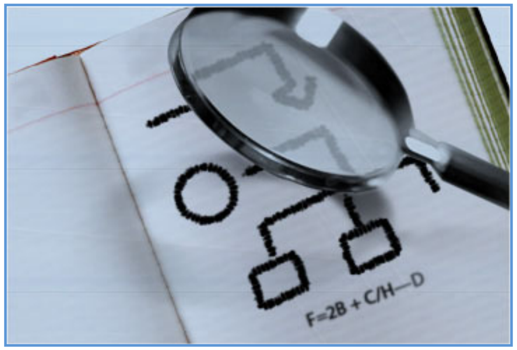
- A naming conflict generally occurs when a server instance attempts bind a service to the JNDI when that name already exists.
  - The services are non-clustered
- In a cluster, a naming conflict can occur when the server tries to bind a clustered object with the same name as a non-clustered object.
  - The object will be bound to the JNDI locally
  - Other servers will not bind the replica-aware stub
- To avoid naming conflicts:
  - Deploy all clusterable objects to all servers in the cluster
  - Deploy to the cluster itself, not the individual servers

Section Review

In this section, we learned how to:

✓ Understand how JNDI clustering works

✓ Deal with JNDI naming conflicts



Road Map

1. Clustering Services
2. JNDI
3. **JDBC**
   - JDBC Clustering
   - Clustering vs. Multi-Data Sources
   - Targeting a Data Source to a cluster
4. Transactions
5. JMS

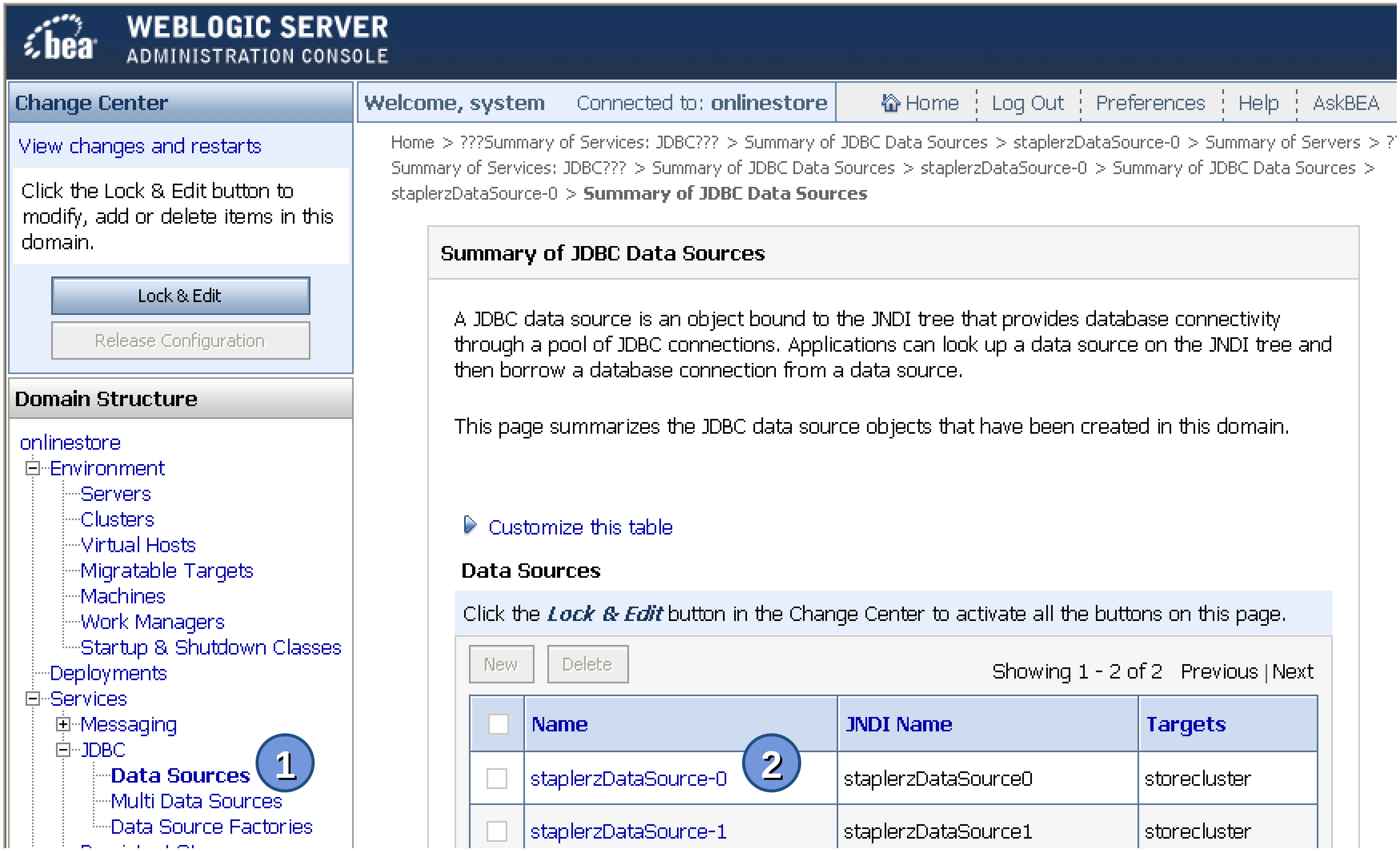
JDBC Clustering

- These JDBC objects are clusterable:
  - Data Sources
  - Multi Data Sources
- When you target a JDBC data source, a new instance of the data source is created on the target.
  - Single server target—a data source instance is created on the server.
  - Cluster target—a data source instance is created on all member servers in the cluster.
- Clustering your JDBC objects does not enable failover of connections but it can ease the process of reconnection when a connection fails.

Clustering vs. Multi Data Sources

Feature	Cluster	Multi Data Source
Local Clients	Access via JNDI	Access via JNDI
Remote Clients	Access via JNDI	Access via JNDI
Failover	None	On the same server (*High-Availability Algorithm only)*
Load balance	None	On the same server *(Load-Balance Algorithm only)*

Targeting a DataSource to a Cluster...



Click the Lock & Edit button to modify, add or delete items in this domain.

Home > ???Summary of Services: JDBC??? > Summary of JDBC Data Sources > staplerzDataSource-0 > Summary of Servers > ? Summary of Services: JDBC??? > Summary of JDBC Data Sources > staplerzDataSource-0 > Summary of JDBC Data Sources > staplerzDataSource-0 > **Summary of JDBC Data Sources**

Summary of JDBC Data Sources

A JDBC data source is an object bound to the JNDI tree that provides database connectivity through a pool of JDBC connections. Applications can look up a data source on the JNDI tree and then borrow a database connection from a data source.

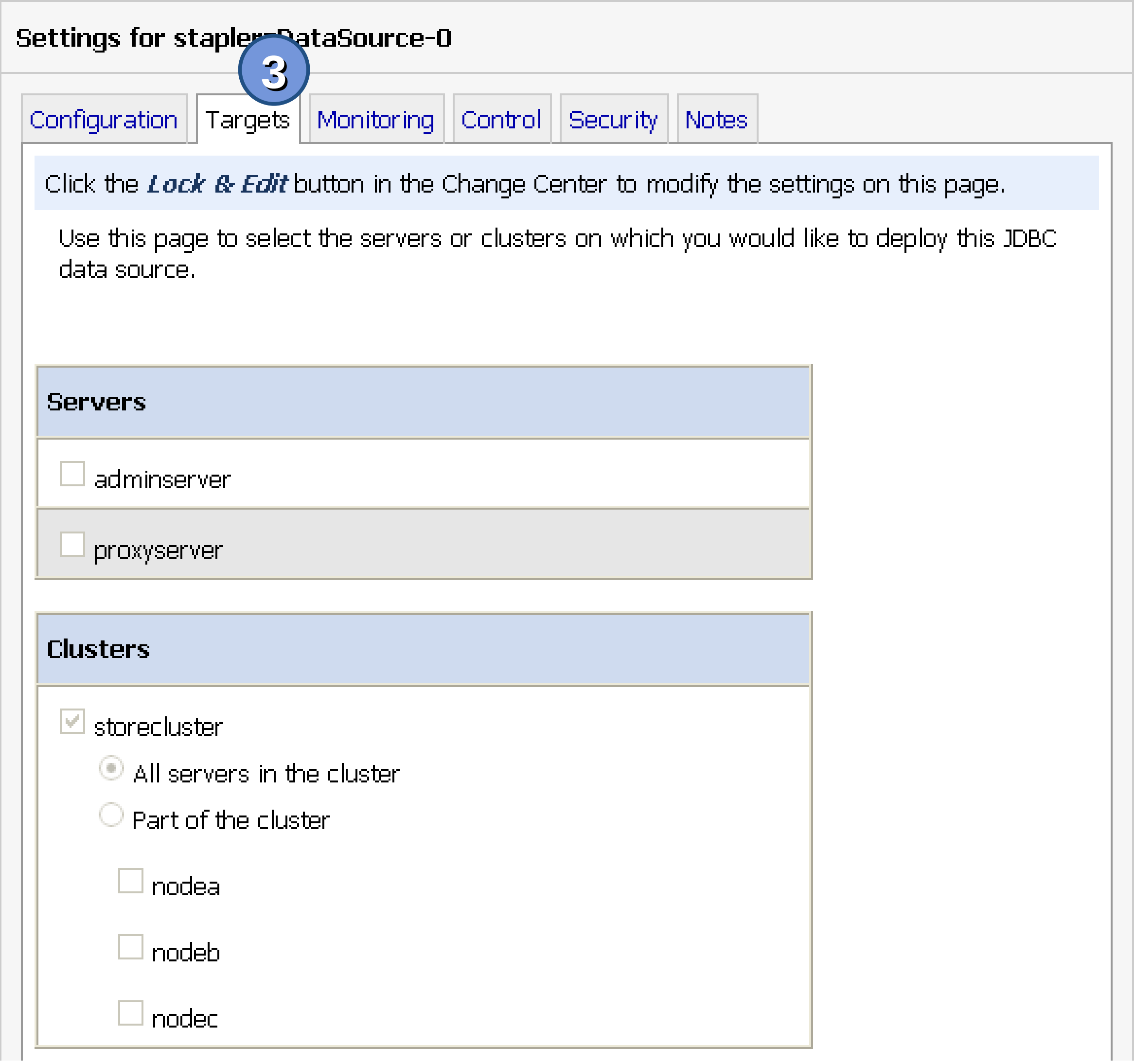
This page summarizes the JDBC data source objects that have been created in this domain.

Data Sources

Click the ***Lock & Edit*** button in the Change Center to activate all the buttons on this page.

□	Name	JNDI Name	Targets
□	staplerzDataSource-0	staplerzDataSource0	storecluster
□	staplerzDataSource-1	staplerzDataSource1	storecluster

...Targeting a DataSource to a Cluster



Settings for stapler-DataSource-0

Configuration | Targets | Monitoring | Control | Security | Notes

Click the ***Lock & Edit*** button in the Change Center to modify the settings on this page.

Use this page to select the servers or clusters on which you would like to deploy this JDBC data source.

Servers

□ adminserver

□ proxyserver

Clusters

☑ storecluster

● All servers in the cluster

○ Part of the cluster

□ nodea

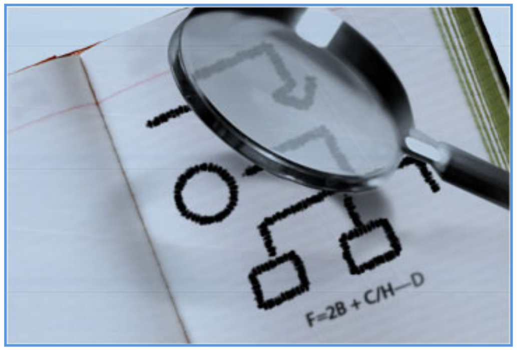
□ nodeb

□ nodec

Section Review

In this section, we learned how to:

- ✓ Understand JDBC clustering capabilities
- ✓ Understand the difference between JDBC clustering and multi data sources
- ✓ Deploy a JDBC data source to a cluster



Exercise

Cluster JDBC Data Sources

- For details on the exercise, refer to the Lab Guide.
- If questions arise, ask the instructor.
- The instructor will determine the stop time.

Road Map

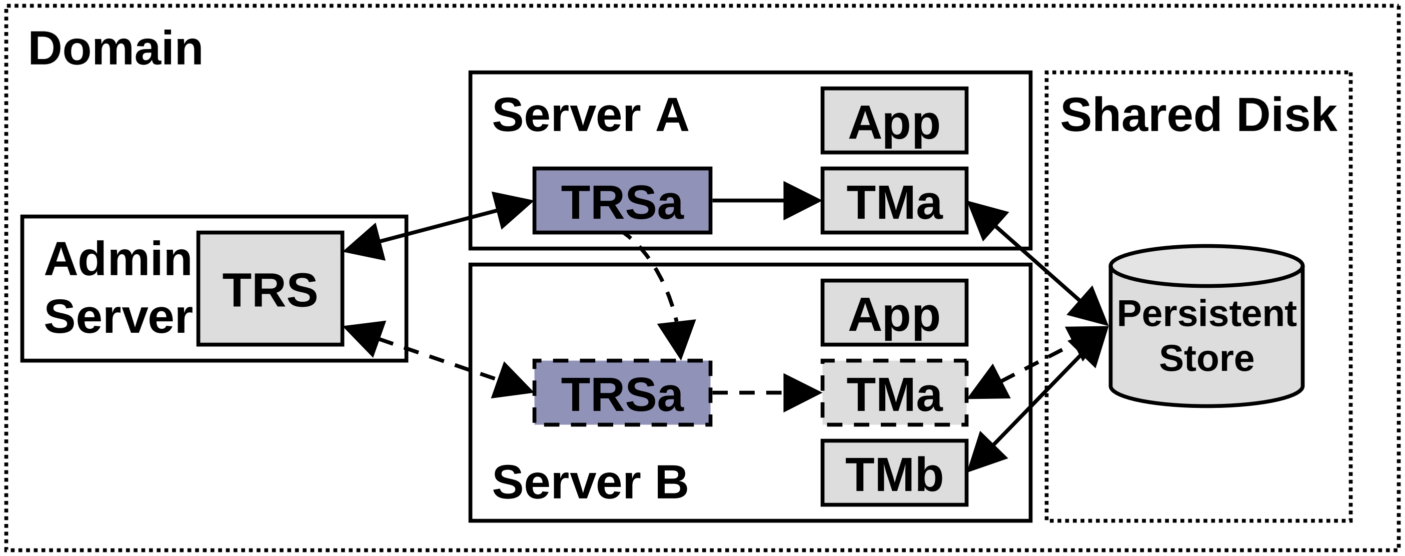
1. Clustering Services
2. JNDI
3. JDBC
4. **Transactions**
   - Transaction Recovery Service
   - Migrating JTA
5. JMS

Transaction Recovering After Failure

- The Transaction Manager (TM) makes every effort to resolve prepared transactions even after multiple crashes.
- The TM uses the Transaction Recovery Service (TRS) which:
  - automatically attempts to recover transactions on system startup
  - owns the transaction log for a server
  - parses all log files on startup for incomplete transactions and completes them

Transaction Recovery Service

- When a server restarts after a crash or if the JTA is migrated, the TRS attempts to:
  - resolved prepared transactions in T-Logs
  - maintain consistency across resources
  - persist in achieving transaction resolution
  - report heuristic completions



Recovering JTA Without a Cluster

- To recover transactions from a non-clustered server:
  - Make the persistent store for the failed server available to the new server
  - Set the path for the default persistent store to the path to the data file
  - Start the new server
- The TRS searches all transaction log files for incomplete transactions.

Recovering Transactions in a Cluster

- When recovering transaction in a clustered server you have two options:
  - Transaction Recovery Service migration
    - the TRS on backup server takes ownership of transaction log from crashed server
    - the TRS searches all transaction log files from failed server and attempts to complete in-flight transactions
    - if TRS on the backup server successfully completes all incomplete transactions, the server releases ownership of the TRS
  - Whole Server Migration
    - migrates the server in its entirety, along with all the services it hosts

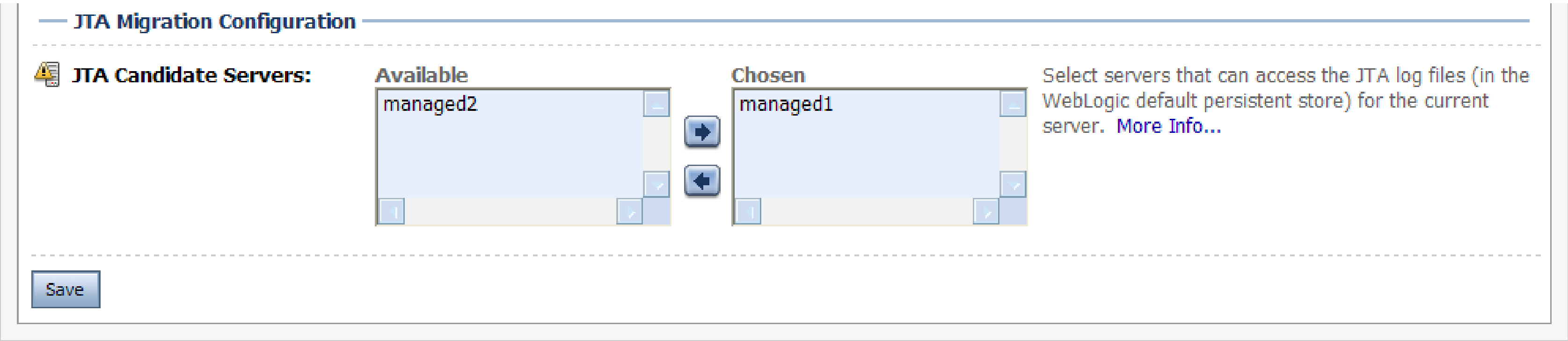
Recovering Transactions for a Failed Clustered Server (Manually)

- Manually migrate the TRS from the crashed server to another server in the cluster using the Admin Console or the command-line interface:
  1. The TRS on the backup server takes ownership of the transaction log from the failed server.
  2. The TRS searches transaction log files from the failed server for incomplete transactions and completes them.
  3. If the TRS on the backup server successfully completes all incomplete transactions from the failed server, it releases ownership of the TRS (including transaction log files) for the failed server, so the failed server can reclaim it upon restart.

Configuring JTA Service Migration

- Before you can manually migrate JTA, you can to define which servers are available for migration.
- To set up the candidate servers use:

  Environment → Servers → Configuration → Migration



JTA Migration Configuration

JTA Candidate Servers:

Available: managed2

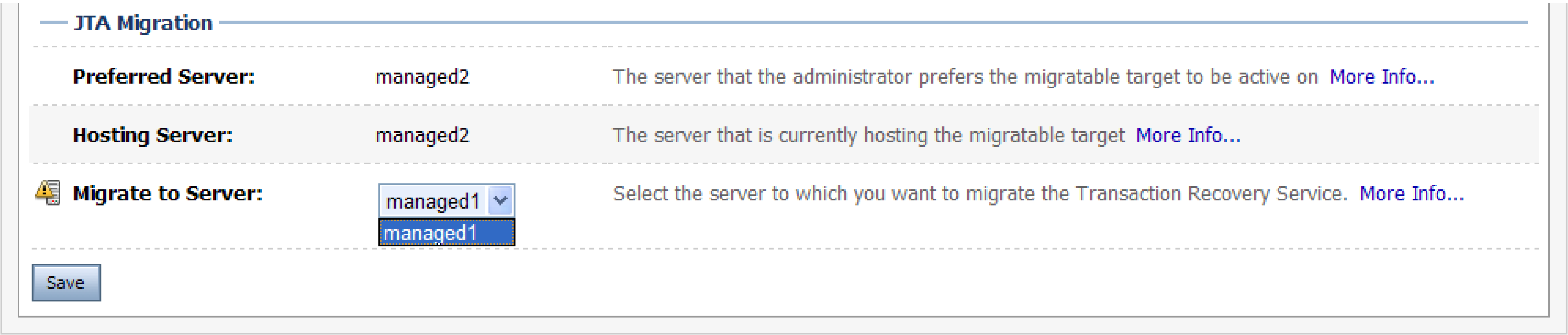
Chosen: managed1

Select servers that can access the JTA log files (in the WebLogic default persistent store) for the current server. More Info...

Manually Migrating the JTA to Another Server in a Cluster

- When you need to migrate transactions from a failed server to a working server use:

Environment → Servers → Control → Migration



JTA Migration

Preferred Server:	managed2	The server that the administrator prefers the migratable target to be active on More Info...
Hosting Server:	managed2	The server that is currently hosting the migratable target More Info...
Migrate to Server:	managed1	Select the server to which you want to migrate the Transaction Recovery Service. More Info...

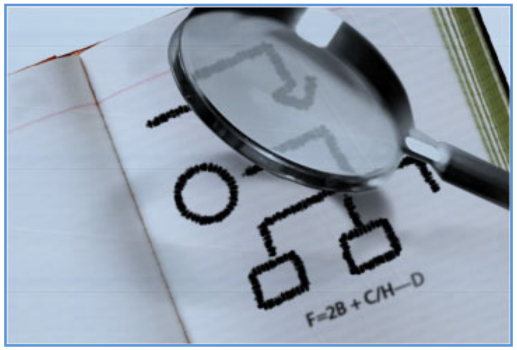
JTA Migration Limitations

- Migrating the JTA has the following limitations:
  - The JTA cannot be migrated to a backup server from a running server
    - The server must be stopped before migrating the JTA
  - The backup server only processes incomplete transactions
    - The backup server does not accept new transaction work for the failed server
  - The backup server does not process heuristic log files

Section Review

In this section, we learned how to:

- ✓ Understand how the transaction recovery service recovers transactions after a failure
- ✓ Manually migrate transactions to another server in the cluster



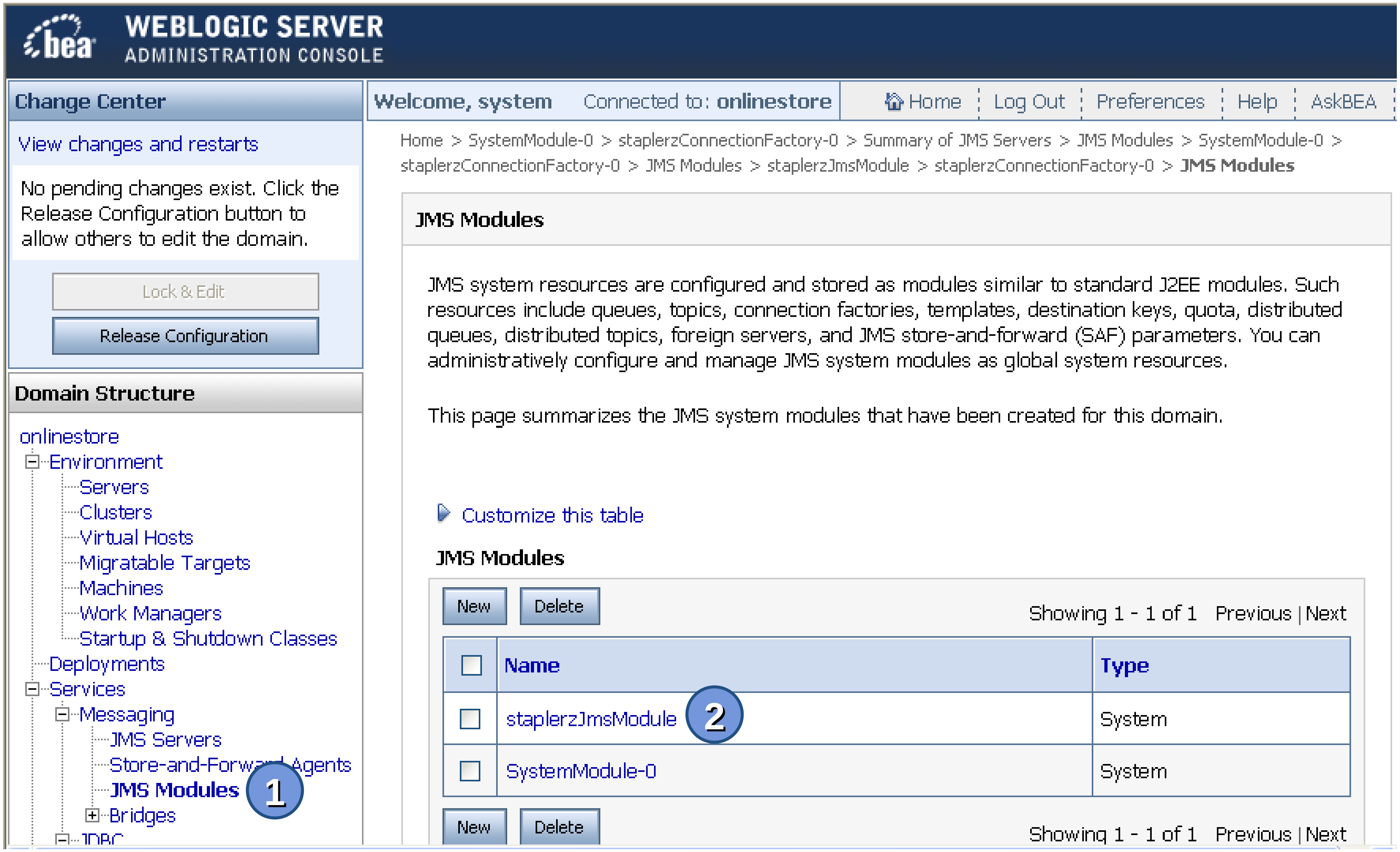
Road Map

1. Clustering Services
2. JNDI
3. JDBC
4. Transactions
5. **JMS**
   - Clustering JMS
   - Distributed Destinations
   - JMS Failover

JMS Clustering Support

- WebLogic Server supports targeting JMS connection factories to a cluster.
- WebLogic Server supports distributing JMS destinations throughout a cluster (Distributed Destinations)
  - JMS Queues and Topics are still managed by a single server instance in the cluster.
- WebLogic Server provides fail-over for JMS messages through:
  - Distributed destinations
  - Server-level automatic migration

JMS Connection Factory Clustering...



No pending changes exist. Click the Release Configuration button to allow others to edit the domain.

Home > SystemModule-0 > staplerzConnectionFactory-0 > Summary of JMS Servers > JMS Modules > SystemModule-0 > staplerzConnectionFactory-0 > JMS Modules > staplerzJmsModule > staplerzConnectionFactory-0 > **JMS Modules**

JMS Modules

JMS system resources are configured and stored as modules similar to standard J2EE modules. Such resources include queues, topics, connection factories, templates, destination keys, quota, distributed queues, distributed topics, foreign servers, and JMS store-and-forward (SAF) parameters. You can administratively configure and manage JMS system modules as global system resources.

This page summarizes the JMS system modules that have been created for this domain.

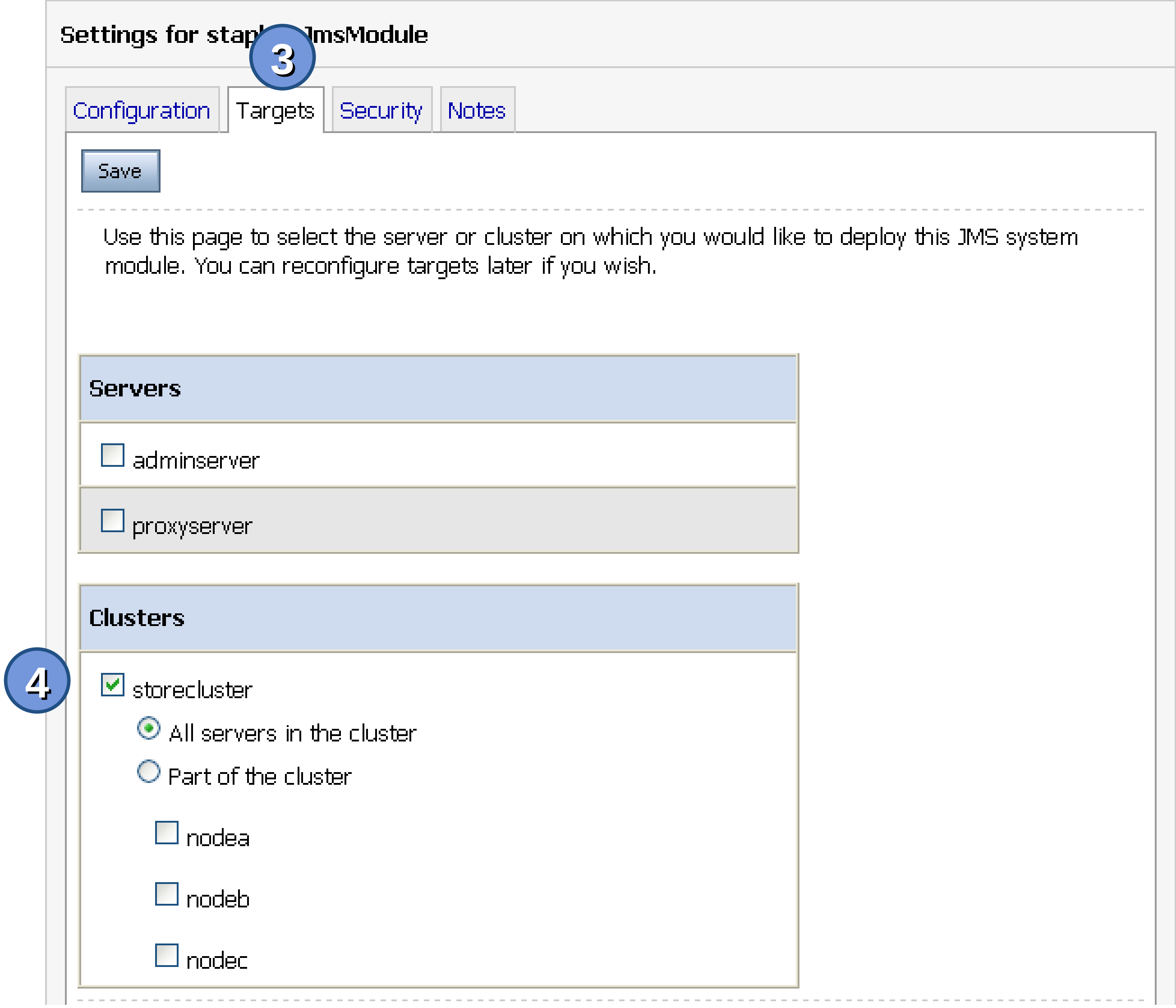
JMS Modules

Showing 1 - 1 of 1 Previous | Next

□	Name	Type
□	staplerzJmsModule	System
□	SystemModule-0	System

Showing 1 - 1 of 1 Previous | Next

...JMS Connection Factory Clustering



Settings for stapl JmsModule

Use this page to select the server or cluster on which you would like to deploy this JMS system module. You can reconfigure targets later if you wish.

Servers

- □ adminserver
- □ proxyserver

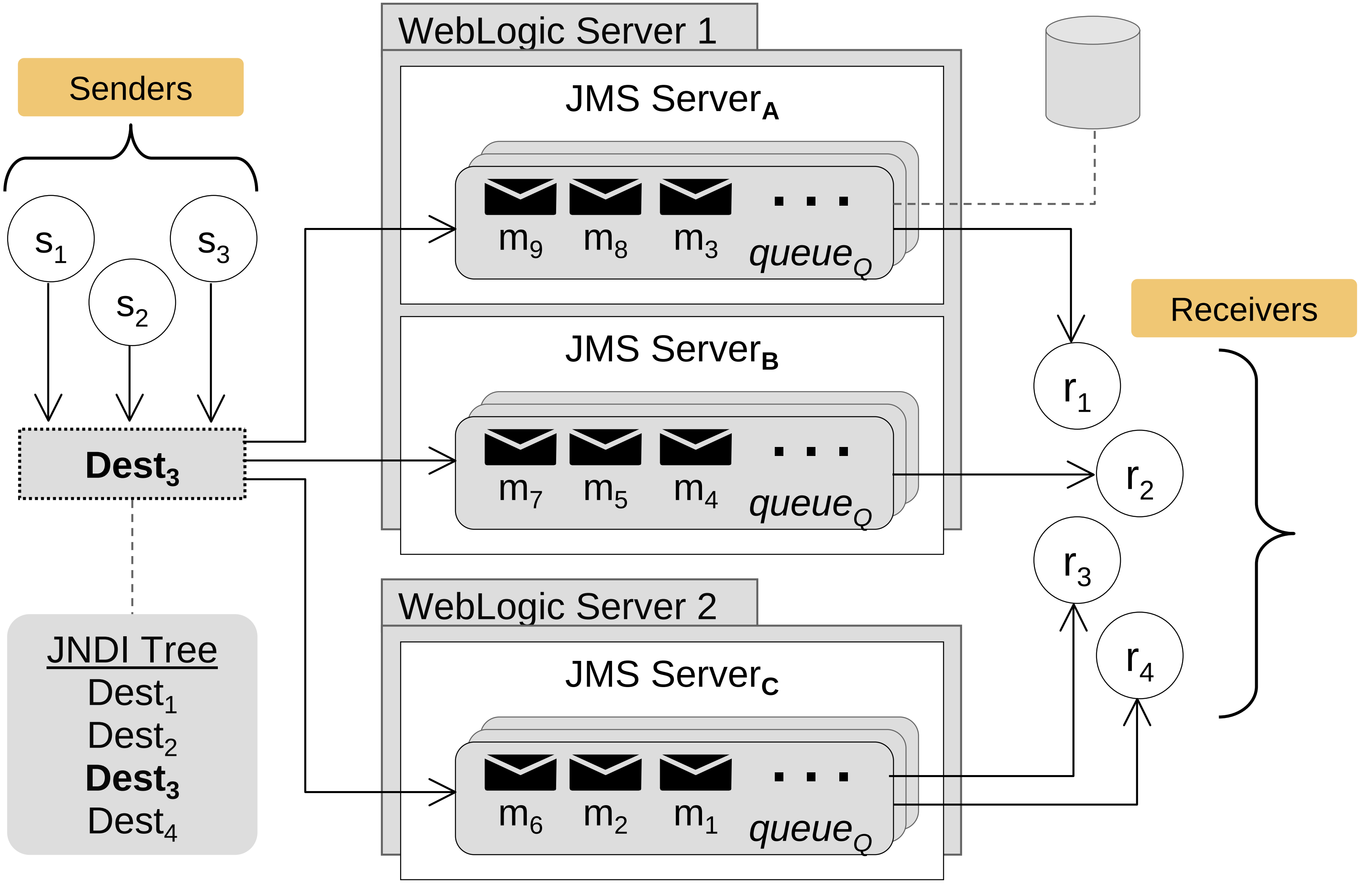
Clusters

- ☑ storecluster
  - ● All servers in the cluster
  - ○ Part of the cluster
    - □ nodea
    - □ nodeb
    - □ nodec

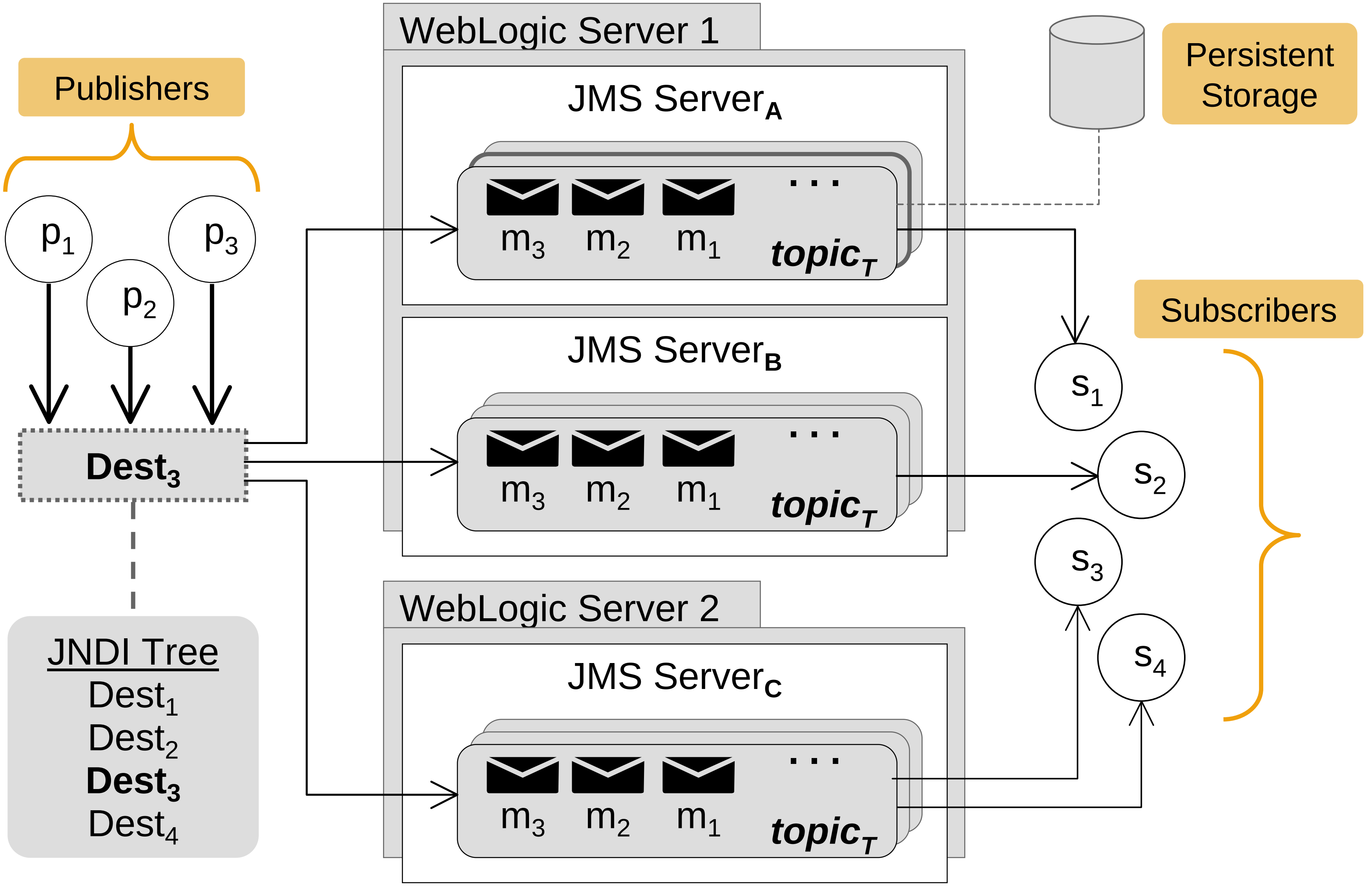
Distributed Destination

- A distributed destination has these properties:
  - It defines multiple destinations as part of one distributed destination.
  - It is looked up as a regular destination, via JNDI.
  - Its member availability is dynamically updated.
- Producers and consumers can send to and receive from a distributed destination.
- WebLogic JMS distributes the load across the available physical destinations within the distributed destination.

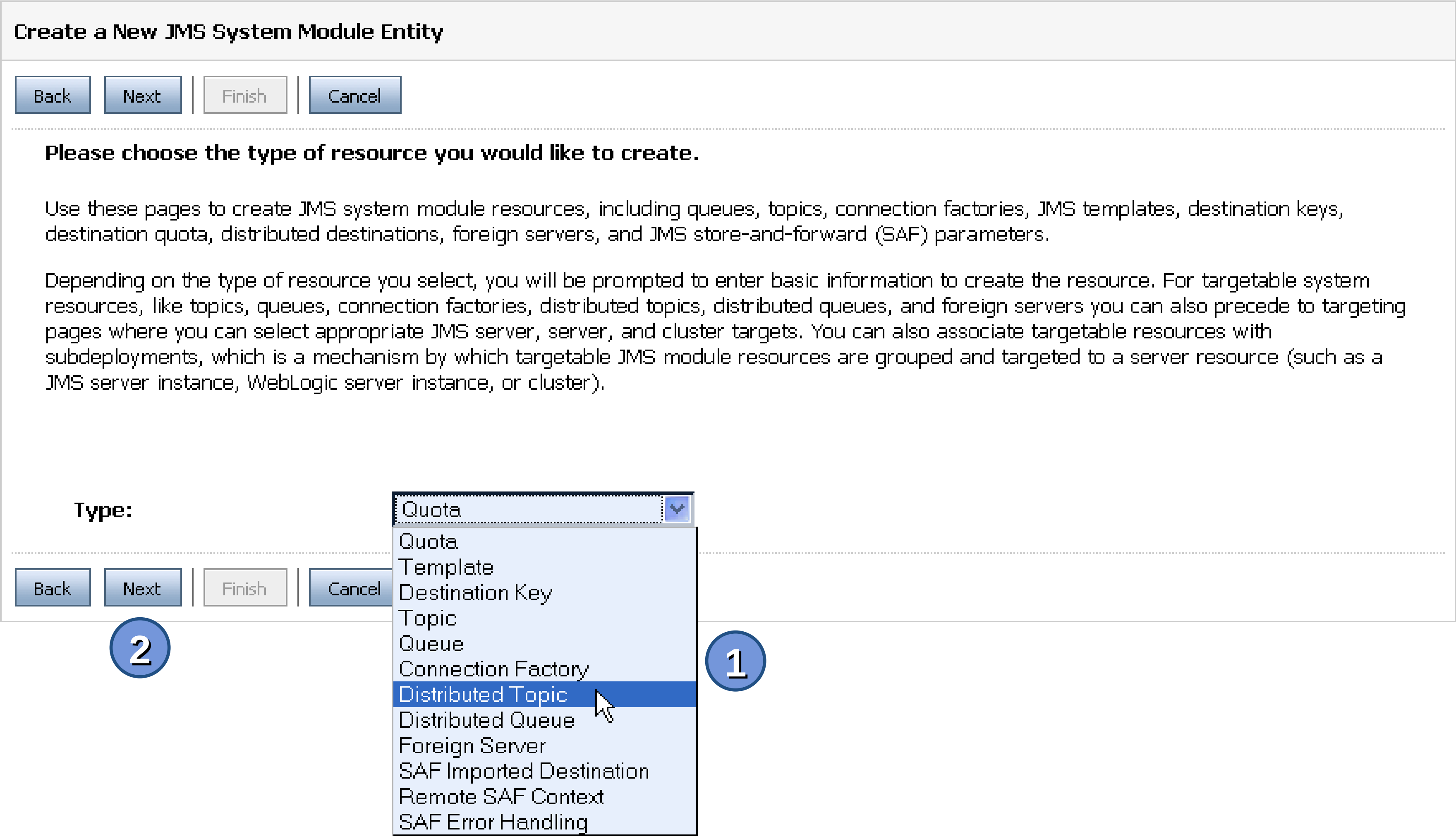
Distributed Queues



Distributed Topics



Create a Distributed Topic…



Create a New JMS System Module Entity

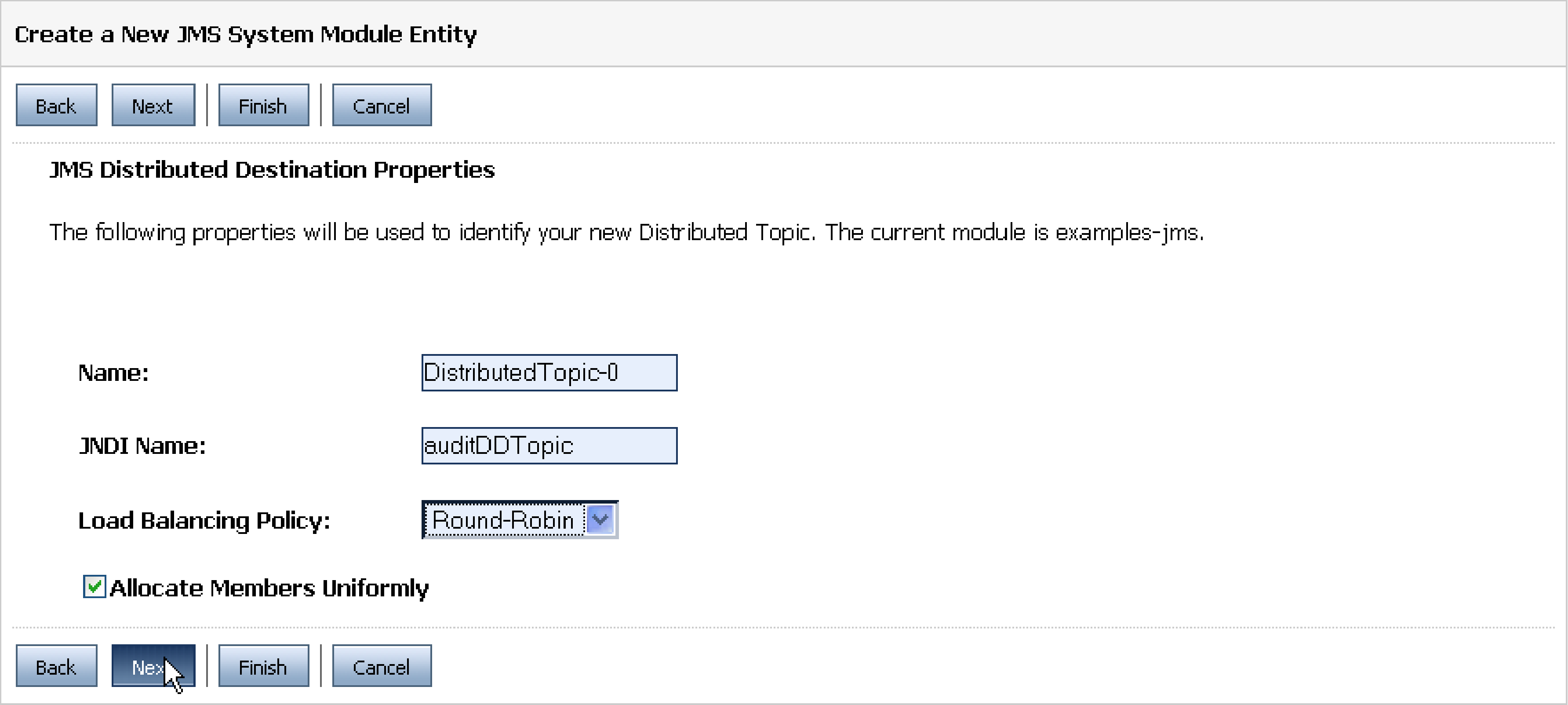
Please choose the type of resource you would like to create.

Use these pages to create JMS system module resources, including queues, topics, connection factories, JMS templates, destination keys, destination quota, distributed destinations, foreign servers, and JMS store-and-forward (SAF) parameters.

Depending on the type of resource you select, you will be prompted to enter basic information to create the resource. For targetable system resources, like topics, queues, connection factories, distributed topics, distributed queues, and foreign servers you can also precede to targeting pages where you can select appropriate JMS server, server, and cluster targets. You can also associate targetable resources with subdeployments, which is a mechanism by which targetable JMS module resources are grouped and targeted to a server resource (such as a JMS server instance, WebLogic server instance, or cluster).

Type: Quota

…Create a Distributed Topic…



Create a New JMS System Module Entity

JMS Distributed Destination Properties

The following properties will be used to identify your new Distributed Topic. The current module is examples-jms.

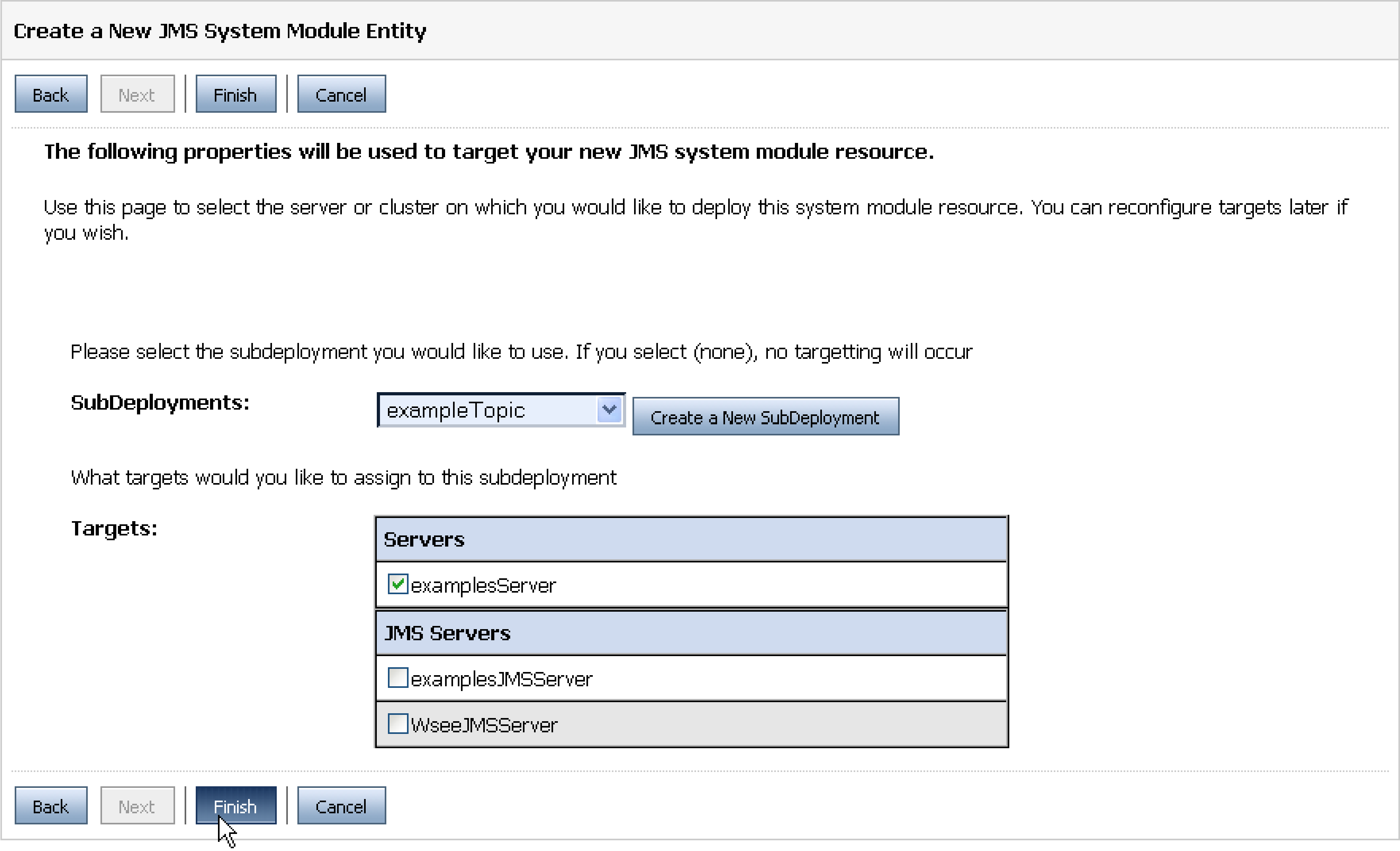
Name: DistributedTopic-0

JNDI Name: auditDDTopic

Load Balancing Policy: Round-Robin

☑ Allocate Members Uniformly

...Create a Distributed Topic



Create a New JMS System Module Entity

The following properties will be used to target your new JMS system module resource.

Use this page to select the server or cluster on which you would like to deploy this system module resource. You can reconfigure targets later if you wish.

Please select the subdeployment you would like to use. If you select (none), no targetting will occur

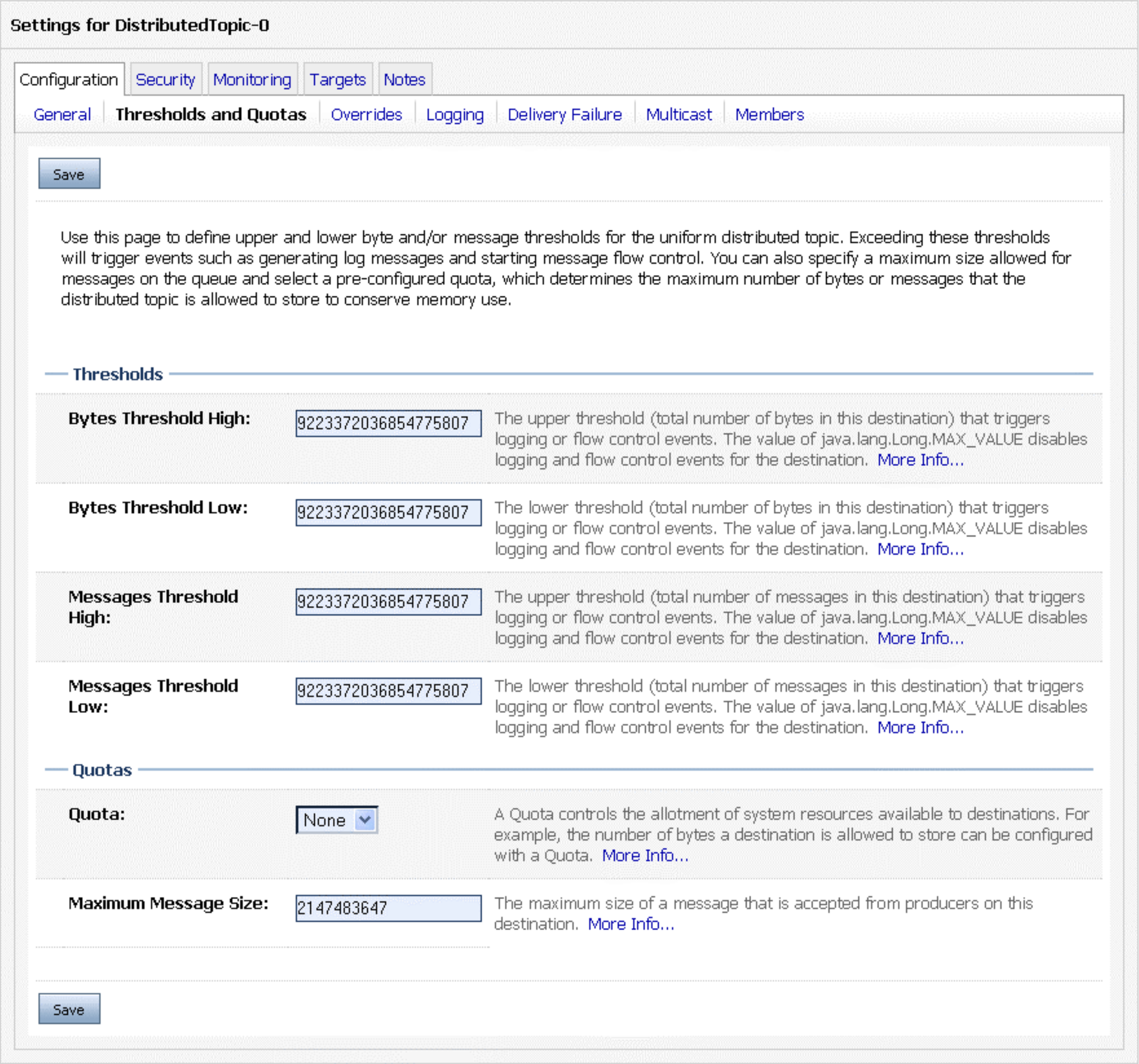
SubDeployments: exampleTopic

What targets would you like to assign to this subdeployment

Targets:

Servers
☑ examplesServer
JMS Servers
□ examplesJMSServer
□ WseeJMSServer

Distributed Destination Threshold and Quota



Use this page to define upper and lower byte and/or message thresholds for the uniform distributed topic. Exceeding these thresholds will trigger events such as generating log messages and starting message flow control. You can also specify a maximum size allowed for messages on the queue and select a pre-configured quota, which determines the maximum number of bytes or messages that the distributed topic is allowed to store to conserve memory use.

Thresholds

Bytes Threshold High:	9223372036854775807	The upper threshold (total number of bytes in this destination) that triggers logging or flow control events. The value of java.lang.Long.MAX_VALUE disables logging and flow control events for the destination. More Info...
Bytes Threshold Low:	9223372036854775807	The lower threshold (total number of bytes in this destination) that triggers logging or flow control events. The value of java.lang.Long.MAX_VALUE disables logging and flow control events for the destination. More Info...
Messages Threshold High:	9223372036854775807	The upper threshold (total number of messages in this destination) that triggers logging or flow control events. The value of java.lang.Long.MAX_VALUE disables logging and flow control events for the destination. More Info...
Messages Threshold Low:	9223372036854775807	The lower threshold (total number of messages in this destination) that triggers logging or flow control events. The value of java.lang.Long.MAX_VALUE disables logging and flow control events for the destination. More Info...

Quotas

Quota:	None	A Quota controls the allotment of system resources available to destinations. For example, the number of bytes a destination is allowed to store can be configured with a Quota. More Info...
Maximum Message Size:	2147483647	The maximum size of a message that is accepted from producers on this destination. More Info...

Server Affinity

- WebLogic will try to establish new connections to the same servers as existing connections.
- Applies to distributed destinations only.

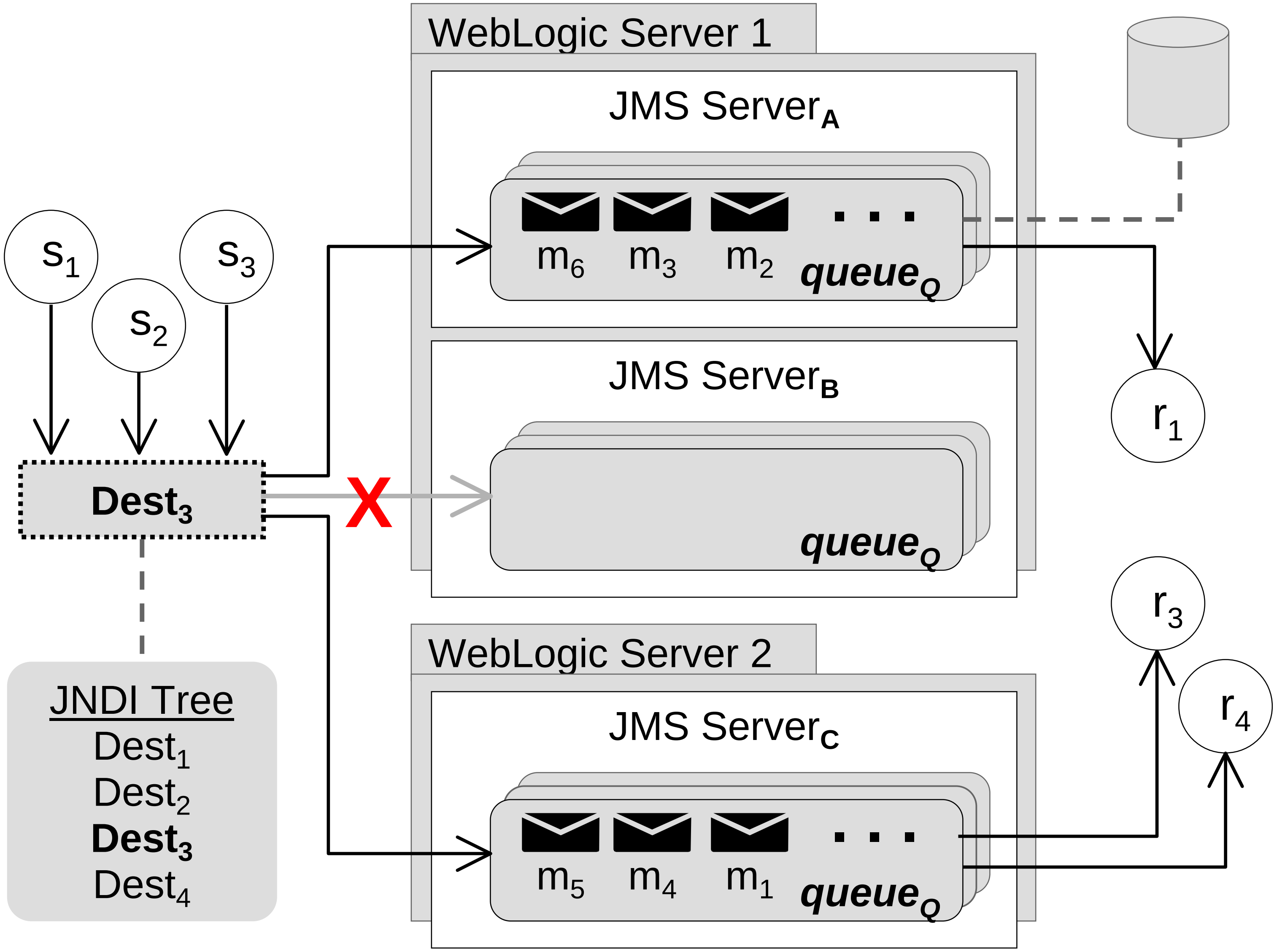
For a new JMS connection distributed destination:

1. Attempts to create a connection on the same server as the JNDI initial context, the original client connection server instance.
2. Attempts to create a connection to a server with which the client already has a connection, JMS or non-JMS. This may be JTA or JMS transaction dependent.
3. Creates the connection on any server.

Zero Consumers…

- *Server affinity* favors members on the same server as the client.
- *Transaction affinity* favors members that are already involved in a transaction.
- A *zero consumer* is a member that has no consumers; it is used as follows:
  - A producer avoids a zero consumer
  - A consumer searches a for zero consumer
- *Persistent messages* are put in a backing store.

…Zero Consumers



Load Balancing

- If the destination is not distributed:
  - All production is directed at one destination
  - The consumer uses that destination
  - Load balancing does not occur
- If the destination is distributed:
  - Production is replicated across member topics and queues
  - Load balancing is either round-robin or random
  - The consumer is pinned to one member destination, which is selected at creation time

Location Transparency

- Standard destinations can be replicated in the cluster-wide JNDI service.
- Choose *Replicate JNDI Name in Cluster* in the Admin console.
- Use the **`isJNDINameReplicated()`** method in the **`JMSQueueMBean`** or **`JMSTopicMBean`**.
- Clients look up a destination on any server and receive a connection on the actual server.

Failover

- Failover is both automatic and manual.
- A manual failover includes migrating these elements:
  - An entire JMS Server
  - A persistent store
  - A transaction log
- WLS provides a migratable JMS service that attempts to deliver outstanding JMS messages.

JMS Server Migration

- JMS is an *exactly-once* service; each JMS server exists on exactly one WebLogic server.
- When a WebLogic server fails, its JMS servers can be migrated. Migration must be configured ahead of time.
- For persistent messaging, migratable targets must have access to the same JMS store as the original server.
- A JMS server can migrate to a WebLogic server that already hosts distributed destination members.
- Migration may be a part of scheduled maintenance.

Performing Migration

- Services migrated to a non-running server will be started when the server starts.
- Migration must can manually initiated or automatically initiated.
  - Automatic migration happens through whole server migration only
- It is possible to use third-party products to perform migration:
  - JMX
  - Veritas HA

JMS Migratable Targets

- The JMS server can be manually migrated to any other server in a cluster
  - Optionally, an administrator can limit which servers will receive the migrated JMS server by specifying a defined set of migratable targets within a cluster.

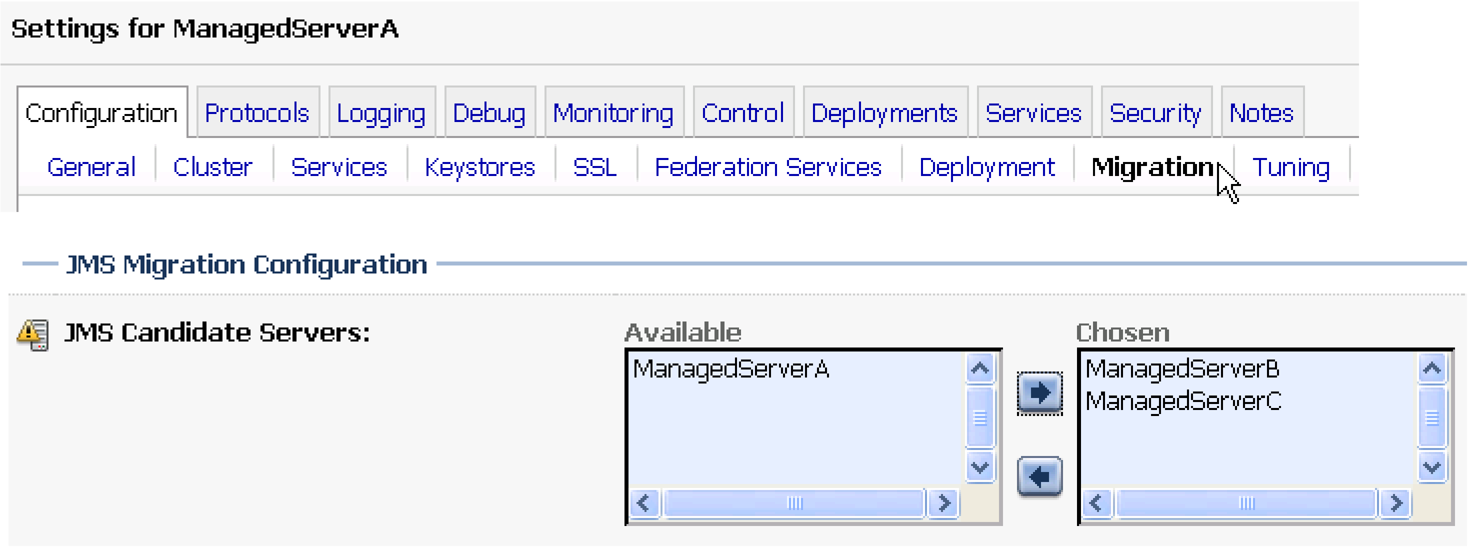
Migrate JMS Data

- For the JDBC store:
  - If it is on the failed server, migrate the database to the new server and change the JDBC URL for the JDBC store's DataSource.
  - If it is not on the failed server, no changes are required.
- For the file store:
  - Migrate to a new server
  - Ensure that the pathname is the same on the new server as the original one
- For transactions, also migrate the transaction logs

Migration Configuration

- To configure a JMS migration:
  1. Define migratable target servers in the cluster.
  2. Create JMS Servers and assign to the migratable targets.
  3. Create *physical* member destinations (use the auto-deploy option for distributed destinations).

Configure a Migratable Target



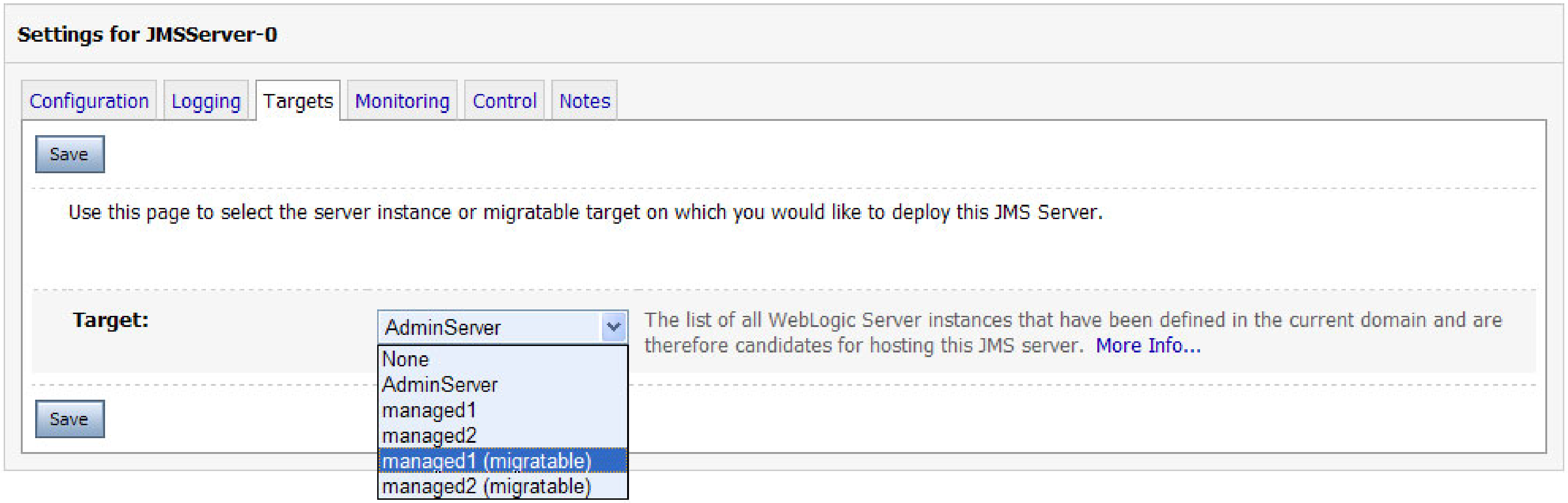
Settings for ManagedServerA

JMS Migration Configuration

JMS Candidate Servers:

Available	Chosen
ManagedServerA	ManagedServerB ManagedServerC

Target JMS Server to a Migratable Target



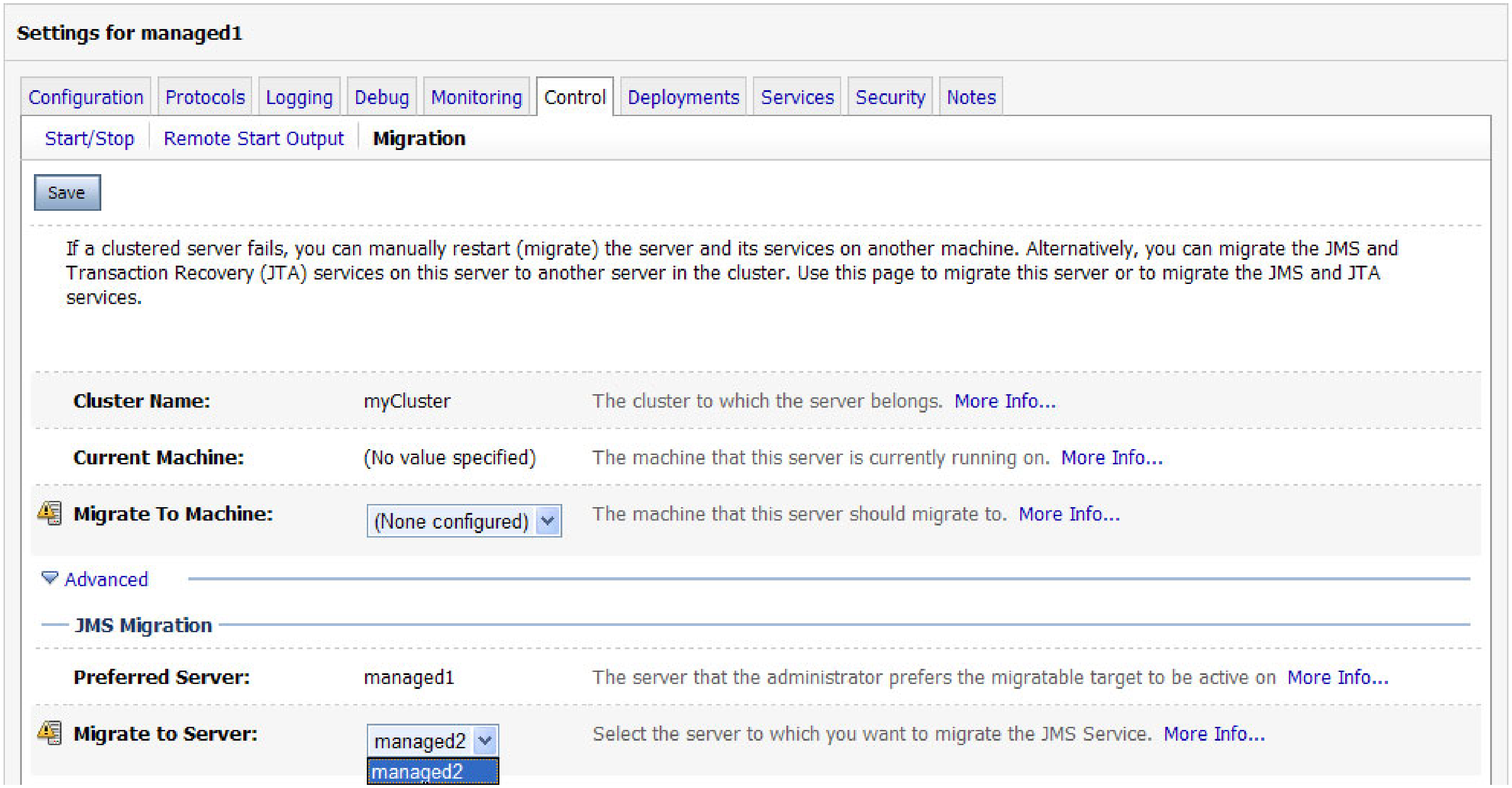
Settings for JMSServer-0

Use this page to select the server instance or migratable target on which you would like to deploy this JMS Server.

Target: AdminServer

The list of all WebLogic Server instances that have been defined in the current domain and are therefore candidates for hosting this JMS server. More Info...

Migrate Services



Settings for managed1

If a clustered server fails, you can manually restart (migrate) the server and its services on another machine. Alternatively, you can migrate the JMS and Transaction Recovery (JTA) services on this server to another server in the cluster. Use this page to migrate this server or to migrate the JMS and JTA services.

Cluster Name:	myCluster	The cluster to which the server belongs. More Info...
Current Machine:	(No value specified)	The machine that this server is currently running on. More Info...
Migrate To Machine:	(None configured)	The machine that this server should migrate to. More Info...

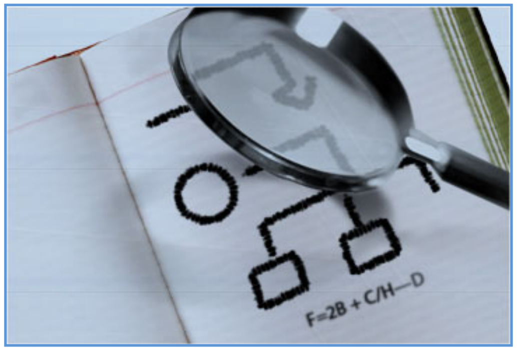
JMS Migration

Preferred Server:	managed1	The server that the administrator prefers the migratable target to be active on More Info...
Migrate to Server:	managed2 managed2	Select the server to which you want to migrate the JMS Service. More Info...

Section Review

In this section, we learned how to:

- ✓ Handle failover and load balancing of JMS connections
- ✓ Cluster a JMS connection factory



Exercise

Load Balance JMS Messages

- For details on the exercise, refer to the Lab Guide.
- If questions arise, ask the instructor.
- The instructor will determine the stop time.

Module Review

In this module, we learned how to:

- ✓ Migrate a whole server
- ✓ Cluster JNDI objects
- ✓ Cluster JDBC data sources
- ✓ Cluster transactions
- ✓ Cluster JMS connection factories