# Protection of the JMS destination

## Goals

**After completing this section, a system administrator should be able to do the following:**

- **Configure role-based access to topics and queues in the messaging subsystem.**

## Security domains for messaging

**It was already explained in the Configuring the messaging subsystem chapter that the default values for the EAP 7 MOM embedded in the messaging activemq subsystem have the following characteristics:**

- **Authentication is required only for remote connections.**
- **JMS connections are authenticated using the other security domain.**
- **All local applications have access to publish and consume messages in all the destinations.**

**This default configuration allows developers to rapidly deploy JMS applications and test them without worrying about security, and without leaving the EAP instance vulnerable to remote attacks. It also allows for easy testing of remote JMS applications, as the developer only needs to add users and roles to the ApplicationRealm security realm using the add-user.sh script.**

**Most production environments probably require three changes:**

- **Allow authentication for local applications.**
- **Change the ActiveMQ security domain to one based on a database relational or an LDAP directory.**
- **Remove the permissions granted to the guest role.**

**These changes are implemented using the following commands:**

- **To allow authentication for local applications, change the override-in attribute vm-security a false:**

```
/subsystem=messaging:/server=default:write-attribute(\ name=override-in-
vm-security,value=false)
```

- **To change the ActiveMQ security domain, modify the security-domain attribute to indicate the desired security domain name. For example, to use the security domain named production-sd:**

```
/subsystem=messaging:/server=default:write-attribute(\ name=security-
doman,value=production-sd)
```

- **After making any of the above configuration changes, reload the server. In domain managed mode, reload all server instances using the modified profile.**

- **There are two ways to remove the permissions granted to the guest role:**

  ÿ **Remove the guest role from the security-setting object from all targets (#):**

```
/subsystem=messaging:/server=default/security-setting=#/rule=guest:remove
```

  ÿ **Remove the entire security-setting object from all targets (#):**

```
/subsystem=messaging:/server=default/security-setting=#:remove
```

**After making these changes, the administrator must either create security-setting objects that grant the correct permissions to the roles associated with ActiveMQ application users, or add new roles to the existing security-setting object of all targets.**

**It is important to understand that ActiveMQ users and roles are not related to application users and roles. Remember that MOM is a server itself. ActiveMQ users and roles control application access to MOM targets in the same way that database users and roles control application access to database tables.**

**In the same way that the users and roles of a database generally do NOT reflect the users and roles of an application, the users and roles of ActiveMQ will NOT reflect the users and roles of the application. The confusion can be caused by the fact that ActiveMQ uses security domains in the same way that an application does, while databases are unaware of EAP security domains. Remember that the ActiveMQ security domain is completely unrelated to the security domain of an application.**

## Protecting a topic (Topic) or a queue

**The MOM ActiveMQ built into EAP 7 does NOT configure access rules on target projects. Access rules are configured in security-setting objects, which function like the address-setting objects described in the Configuring the messaging subsystem chapter. This makes it easy to configure access rules for applications that need multiple destinations on a consistent basis.**

**A security-setting name is a wildcard expression that matches one or more internal ActiveMQ names of targets. The wildcard does not match the destination JNDI name or the destination object given from the EAP CLI. The easiest way to find a destination name to use as a wildcard is from the queue-address runtime attribute.**

**Another way to find the internal name of a destination is by following the naming convention that ActiveMQ follows to generate the internal name of the destination object name. The internal name is generated by adding a prefix to the name of the target object. The prefixes are:**

- **jms.queue. for jms-queue objects, that is, JMS Queue resources.**

- jms.topic. for jms-topic objects, ie JMS Topic resources.

A security-setting object that matches jms-topic objects also requires a wildcard suffix (either # or *), since JMS Topic resources are internally implemented by ActiveMQ as multiple queues, one per subscriber. Each of those queues is given a suffix based on the subscriber ID.

A security-setting has child objects of type role. The name of this child object is the name of the role that is getting access, and its attributes are permissions. ActiveMQ understands the following permissions that loosely correspond to the JMS API operations that can be performed on a destination:

- send: allows the publication of messages at the destination.
- consume: allows the consumption of messages from the destination.
- createDurableQueue: Allows the creation of durable queues.
- deleteDurableQueue: allows the deletion of durable queues.
- createNonDurableQueue – Allows the creation of temporary queues.
- deleteNonDurableQueue – Allows the deletion of temporary queues.
- manage – Allows management operations on the target, using an API Proprietary ActiveMQ.

There are no specific permissions for JMS Topic resources, as they are implemented by ActiveMQ as a queue for each subscriber. If a client has consume permission on the jms-topic object, they have permission to subscribe to it.

To grant permissions to a target for a role, create the role object inside a security-setting object, whose name matches the target, and set the permission-granted attributes to true. All other permissions will default to false.

For example, the following commands grant send and consume permissions to the publisherAndConsumer role on the jms-topic named StockQuotes:

```
/subsystem=messaging-activemq/server=default/security-setting=\
jms.topic.StockQuotes.#:add()
```

```
/subsystem=messaging-activemq/server=default/security-setting=\
jms.topic.StockQuotes.#/role=publisherAndConsumer:add(\
send=true,consume=true)
```

If multiple security-setting objects match the same target, the most specific one overrides the generic ones. In the above example, if the security-setting of all targets (#) granted manage permission to the publisherAndConsumer role, the jms.topic.StockQuotes.# security-setting denies this.

# Quiz: JMS Destination Protection

**Choose the correct answer to the following questions:**

**1. Which of the following are default characteristics of the subsystem of EAP messaging? (Choose three options).**

a. **JMS connections without set security settings are authenticated using the other security domain.**

b. **All local applications have access to publish and consume messages to all destinations.**

c. **Queues and topics limit manage rights to only users with the admin role.**

**d. By default, authentication is required only from the remote connections.**

It is. **No security domain is used until one is manually configured.**

**2. What security settings allow users to post messages? (Choose One option).**

a. Send b.

Consume Manage

c.

d. CreateDurableQueue

**3. Which of the following pattern values can be used to match a queue named jms.queue.news.europe.fr? (Choose three options).**

a. # b.

jms.queue.news.europe.#

c. jms.queue.news.europe.fr

d. jms.queue.news.europe.fr.#

It is. news.europe.#

**4. What role are given to users by default if no messaging subsystem configurations are done? (Choose one option).**

to. admin b.

No role is given. guest

c.

d. user

# Solution

**Choose the correct answer to the following questions:**

**1. Which of the following are default characteristics of the subsystem of EAP messaging? (Choose three options).**

a. **JMS connections without set security settings are authenticated using the other security domain.**

b. **All local applications have access to publish and consume messages to all destinations.**

c. Queues and topics limit manage rights to only users with the admin role.

**d. By default, authentication is required only from the remote connections.**

**and.** No security domain is used until one is manually configured.

**2. What security settings allow users to post messages? (Choose One option).**

**a. Send b.**

Consume Manage

c.

d. CreateDurableQueue

**3. Which of the following pattern values can be used to match a queue named jms.queue.news.europe.fr? (Choose three options).**

**a. # b.**

**jms.queue.news.europe.#**

c. **jms.queue.news.europe.fr**

d. jms.queue.news.europe.fr.#

It is. news.europe.#

**4. What role are given to users by default if no messaging subsystem configurations are done? (Choose one option).**

**to.** admin **b.**

No role is given. **guest**

c.

d. user

## Password store configuration

## Goals

**After completing this section, a system administrator should be able to do the following:**

- **Protect passwords stored in server configuration files using the password setting.**

EAP **Password Store simplifies much of**

**the server configuration by consolidating all subsystem configurations into a single XML file (standalone.xml or domain.xml). However, this can expose sensitive information to any user who has access to the configuration files. For example, data source credentials are stored by default in plain text in the XML file.**

**The vault encrypts sensitive strings, stores them in an encrypted keystore, and decrypts them for verification and application systems. To prevent sensitive credentials from being readable in the standalone.xml or domain.xml files, users can save passwords or other attributes to the store and then reference it from within the server's configuration file. Using a store creates a layer of abstraction and hides data that could otherwise be read by anyone with access to the configuration files.**

**Creating the password store The process of saving a**

**password to the store is accomplished by the following steps: 1. Create a Java keystore.**

2. **Start the EAP store with the keystore.**

**3. Keep confidential information in the vault.**

**4. Update the server configuration to include the vault information.**

**5. Reference the attribute stored in the server's configuration file.**

**Create a Java keystore to store sensitive strings The store uses a keystore**

**using the certificate stored in the**

**keystore as the encryption key for the entire store. To start the store, users must first create the JavaSE keystore. The following is the syntax used to create a private key and certificate, and to store them in a keystore:**

```
keytool -genseckey -alias <alias> \ -keyalg
<algorithm> -storetype <type> -keysize size \ -keystore <filepath>
```

- **alias – The alias is a unique identifier for the vault or other data stored in the key vault.**

- **keyalg: The algorithm to use the encryption.**
- **storetype: The type of keystore. jceks is the recommended type.**
- **keysize: the size of the encryption key that determines the difficulty of an attack by brute force the key.**
- **keystore – The path and name of the file where the keystore values are stored. of keys.**

```
[student@workstation ~]$ keytool -genseckey -alias vault \ -keyalg AES
-storetype jceks -keysize 128 \ -keystore /home/student/
vault.keystore
```

**After running the command, users will be prompted for a keystore password, and a keystore file will be created at /home/student/vault.keystore.**

## Warehouse use

**The store can be started interactively, supplying each parameter one at a time, or supplying all parameters initially.**

**The following is an example of the syntax used to start the store:**

```
vault.sh --keystore KEYSTORE_URL --keystore-password KEYSTORE_PASSWORD \ --alias
KEYSTORE_ALIAS --vault-block VAULT_BLOCK --attribute ATTRIBUTE \ --sec-attr SEC-ATTR --
enc-dir ENC_FILE_DIR --iteration ITERATION_COUNT \ --salt SALT
```

**To start the store, the following parameters are required:**

- ***KEYSTORE_URL* – The path to the previously created keystore file.**
- ***KEYSTORE_PASSWORD* – The password to access the keystore that was used during keystore creation.**
- ***SALT:* A random eight-character string used to encrypt the attribute stored in the store.**
- ***KEYSTORE_ALIAS:* The alias that identifies the certificate stored in the keystore.**
- ***ITERATION_COUNT* – The number of times the encryption is executed.**
- ***ENC_FILE_DIR* – The path where encrypted files are stored.**
- ***VAULT_BLOCK:* The name to give the block in the vault.**
- ***ATTRIBUTE:* The name of the attribute being stored. For example, "password" as an attribute name when storing a password value.**
- ***SEC-ATTR:* The value that is stored.**

**The following is an example with the parameters filled in:**

```
[student@workstation ~]$ vault.sh --keystore /home/student/vault.keystore \ --keystore-password
password --alias vault --vault-block bookstore \
--attribute password --sec-attr redhat --enc-dir /home/student/ --iteration 50 --salt
12345678
```

**After running the vault.sh command, an XML definition of the vault is displayed, which must be added to the server's configuration files. The following is an example of the XML that needs to be added to the standalone.xml or host.xml configuration file directly before the <management> section:**

```
<vault>
    <vault-option name="KEYSTORE_URL" value="/home/student/vault.keystore"/> <vault-option
    name="KEYSTORE_PASSWORD" value="MASK-31x/z0Xn83H4JaL0h5eK/N"/> <vault-option
    name="KEYSTORE_ALIAS" value="vault"/> <vault-option name="SALT"
    value="12345678"/> <vault-option name="ITERATION_COUNT"
    value="50"/> <vault-option name="ENC_FILE_DIR" value="/home/
    student/"/> </vault>
```

**The final step is to replace the sensitive data with a reference to the attribute in the store. The vault.sh command provides the exact syntax needed to reference the vaulted attribute after supplying all the parameters. Using the example above, the store command generated the following:**

```
VAULT::bookstore::password::1
```

**To use this reference, use the following syntax within the server configuration:**

```
${VAULT::VAULT_BLOCK::ATTRIBUTE_NAME::1}
```

**The following can replace the old password in the server configuration for the bookstore data source password:**

```
${VAULT::bookstore::password::1}
```

**After you replace the password for the data source, the server configuration for the data source will look similar to the following:**

```
<datasource jndi-name="java:jboss/datasources/nookdyotr" ...> <connection-url>...</
                    connection-url> <driver>mysql</driver> <security>


                        <user-name>bkadmin</user-name>
                        <password>${VAULT::bookstore::password::1}</password>
                    </security>
</datasource>
```