Configuring a domain controller

Goals

After completing this section, students should be able to do the following:

- **Describe configuration options for a domain controller and make changes configuration on the domain controller.**

Domain controller values

Domain controller settings are found in two locations: host.xml and domain.xml. The values from the two files are combined to configure and create the domain controller. The domain.xml file has almost the same structure as the standalone.xml file. The differences are described below:

- **standalone.xml can define a single profile, while domain.xml can define any number of different profiles.**
- **domain.xml contains a <server-groups> section for the definition of a common configuration for a group of EAP servers and their assignment to a profile. Because a standalone server is made up of a single server instance, the concept of server groups is meaningless.**

The domain.xml file is structured as follows:

```
<domain xmlns="urn:jboss:domain:4.1">
    <extensions>
        ... domain controller extensions
    </extensions>
    <system-properties> ...
        for defining system properties </system-
    properties> <paths> ... for
    defining
        filesystem paths </paths> <management> ...
    the
    management
        interfaces and their security settings appear here </management> <profiles> <profile


    name="profile_name"> <subsystem
            xmlns="subsystem_xmlns">
                    ... Configuration from subsystems </
            subsystem> </
        profile> </
    profiles>
    <interfaces>
        ... interfaces are defined here </interfaces>
    <socket-binding-
    groups>
        ... Socket binding groups configuration are defined here
    </socket-binding-groups>
    <deployments> ...
        deployments are defined here </
    deployments>
    <server-groups> ...
        server groups are defined here
```

```
    </server-groups> </
domain>
```

Note that the profiles are defined in the <profiles> section and that each <profile> must have a unique name.

The <server-groups> section defines and groups servers, which is not possible in standalone mode. Servers and server groups will be discussed later in this course.

A comparison between domain.xml and host.xml

Both domain.xml and host.xml use a range of XML tags to specify their configuration options. Some of these tags appear in both files. The following table describes which tags appear in which file.

domain.xml o host.xml

configuration name	domain.xml	host.xml
<extensions>	x	x
<system-properties>	x	x
<paths>	x	x
<management>	x	x
<profiles>	x	x
<interfaces>	x	x
<domain-controller>		x
<jvms>		x
<servers>		x
<socket-binding-groups>	x	
<server-groups>	x	
<deployments>	x	

Important The values

in host.xml are combined with the values in domain.xml to configure the host controller. If the same value appears in both files, the value in host.xml takes precedence.

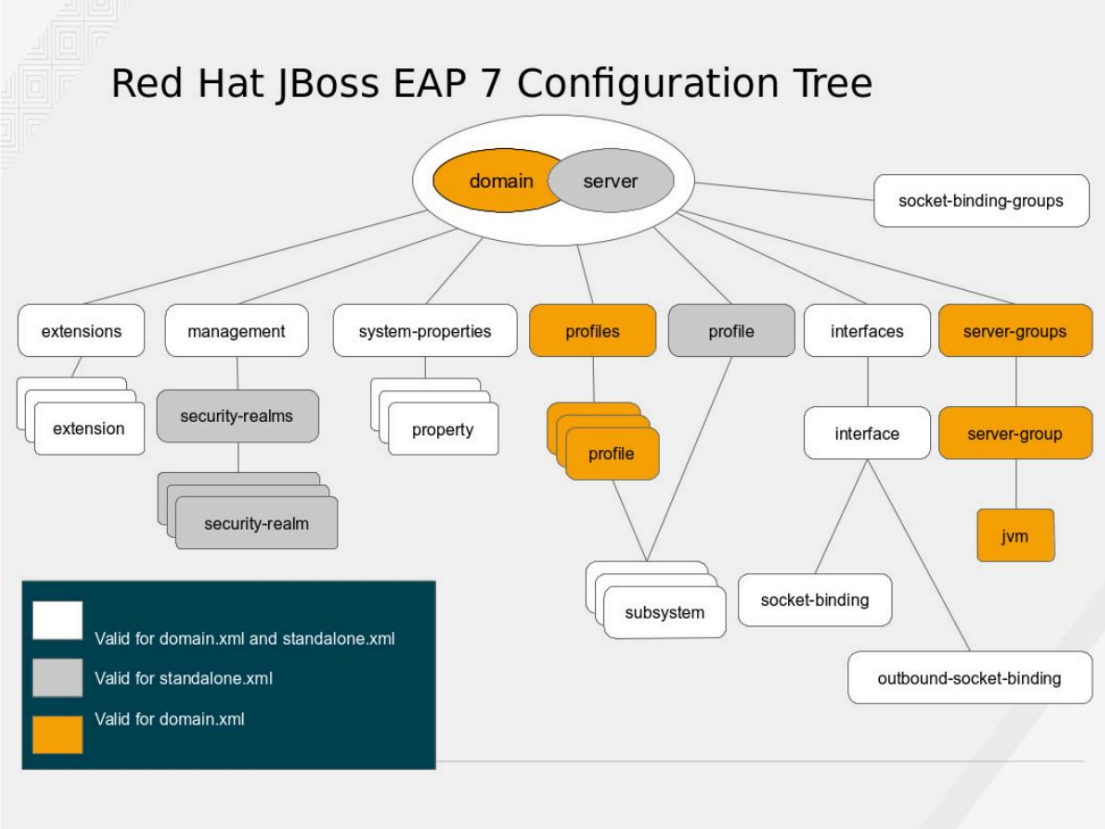

Figure 4.8: XML structure comparison

The deployed applications of a domain are configured in the domain.xml file of the domain controller. This does not mean that every host and server in the domain must implement every application. An application is not deployed to an entire domain, but to a group of servers within the domain. Adding an enterprise application file (EAR *file)* or web application file (WAR *file)* to the <deployments> section simply makes the application available for deployment to a server group.

A <path> defined in domain.xml does not have a value, just a name. The actual value is defined in host.xml. This allows the variable to be defined at the domain level, but allows each host to have a unique value for that variable. This is extremely useful because the hosts may not use the same operating systems or have files deployed in completely different locations.

A definition of <interface> is similar to <path> in domain.xml. When configuring the interfaces of a domain, the domain controller does not need to know the specific addresses of each host, so in domain.xml, the <interface> element works as a placeholder that will be overridden in the host.xml file. of the host controller.

For example, only the interface name is defined in domain.xml:

```
<interface name="my_interface"/>
```

Continuing with the example, the actual IP address or device name for the interface is specified in host.xml:

```
<interface name="my_interface"> <nic
        name="eth1"/> </
interface>
```

Each server is managed by the host controller and not by the domain controller, and each server is declared as part of the host.xml configuration file.

Notice that <server-groups> is defined at the domain level, but <servers> is defined at the host level. This means that the domain.xml and host.xml files must be compatible because a <server> definition in host.xml must reference a <server-group> definition in domain.xml. If a <server> definition in host.xml references a <server-group> that is not defined in domain.xml, the host controller will not start.

Also note that <deployments> is defined at the domain level, not the host level. It is not possible to deploy applications on servers. They must be deployed in groups of servers.

In EAP 7, it is possible to define <extensions> in the host.xml file to support subsystems running on specific hosts.

Configuring a managed domain with the CLI

The way to navigate in domain mode with the CLI is different from standalone mode. New resources are available at the top level:

```
[domain@172.25.250.254:9990 /] cd
```

Using tab completion, the following levels are displayed:

```
-help                                   extension                       profile
--no-validation  core-                    host  server-group interface  socket-binding-group
service                                   management-client-content system-property path
deployment
deployment-overlay
```

The first difference is the host level. This level lists all the host controllers managed by the domain:

```
[domain@172.25.250.254:9990 /] cd host
[domain@172.25.250.254:9990 host] ls host2
host3 master
```

The host tier is used to manage and obtain runtime metrics for available hosts. For example, it is possible to add a new server to host host3:

```
[domain@172.25.250.254:9990 /] cd /host=host2/server-config
[domain@172.25.250.254:9990 server-config] ./server-c:add\ (auto-start=true,
group=main-server-group)
```

To remove a server, use the remove operation:

```
[domain@172.25.250.254:9990 /] /host=host2/server-config/server-c:\ remove()
```

Another important element at the host level is the ability to obtain runtime information. For example, to get memory runtime metrics from a server, use the following commands:

```
[domain@172.25.250.254:9990 /] cd /host=host2/server=server-one/
[domain@172.25.250.254:9990 server=server-one] cd core-service=platform-mbean/
[domain@172.25.250.254:9990 core-service=platform-mbean] ./type=memory:\ read-
attribute(name=heap-memory-usage)
```

A similar result is expected:

```
{
    "outcome" => "success",
    "result" => { "init"
        => 67108864L, "used" =>
        126501384L, "committed"
        => 206045184L, "max" =>
        477626368L
    }
}
```

The top level also provides the server-group level. This level is responsible for creating and managing the server groups. To create a new group, use the following command:

```
[domain@172.25.250.254:9990 core-service=platform-mbean] cd /
[domain@172.25.250.254:9990 /] /server-group=production:add\
(profile=ha,socket-binding-group=ha-sockets)
```

To remove a server group named production, use the remove operation:

```
[domain@172.25.250.254:9990 /] /server-group=production:remove()
```

To remove a server named server-two from a host controller named serverb, use the following operation:

```
[domain@172.25.250.254:9990 /] /host=serverb/server-config=server-two:remove()
```

To configure or manage a subsystem, you must first navigate to the desired named profile:

```
[domain@172.25.250.254:9990 /] cd /profile=ha/subsystem=ejb3
```

The socket-binding-group tier is responsible for configuring the sockets available for server groups to use.

The :take-snapshot operation is available to back up domain.xml configurations. This operation generates a backup file in the JBOSS_HOME/domain/configuration/ domain_xml_history/snapshot folder:

```
[domain@172.25.250.254:9990 /] :take-snapshot
```

Quiz: Configuring a Domain Controller

Choose the correct answer to the following questions:

1. What tags are used in the domain.xml and host.xml files? (Choose three options).

a. extensions b. interfaces

c. servers

d. paths

e. server-groups

2. Which of the following statements are correct? (Choose two options.)

to. domain.xml Configuration definitions take precedence over host.xml b. Host.xml configuration definitions take precedence over domain.xml.

c. The implementations are stored in the host.xml file.

d. The network configuration IP addresses are defined in the host.xml file.

3. Which of the following can be used to create a snapshot of a domain controller configuration? (Choose one option).

a. :take-snapshot :save-

b.

c.

d. snapshot :snapshot :save-snapshot <filename>

Solution

Choose the correct answer to the following questions:

1. What tags are used in the domain.xml and host.xml files? (Choose three options).

a. extensions b. interfaces

c. servers

d. **paths**

e. server-groups

2. Which of the following statements are correct? (Choose two options.)

to. domain.xml Configuration definitions take precedence over host.xml b. Host.xml configuration definitions take precedence over domain.xml.

c. The implementations are stored in the host.xml file.

d. **The network configuration IP addresses are defined in the host.xml file.**

3. Which of the following can be used to create a snapshot of a domain controller configuration? (Choose one option).

a. **:take-**

b. **snapshot :save-**

c.

d. snapshot :snapshot :save-snapshot <filename>