Configuring JBoss EAP as a standalone server

Goals

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

- **Describe the standalone.xml configuration file.**
- **Make configuration changes and upgrades to EAP as a standalone server**

The standalone.xml configuration file

Similar to EAP 6, configuring EAP 7 happens in a single XML file. For a standalone server, the file is JBOSS_HOME/standalone/configuration/standalone.xml. The general structure of standalone.xml looks like this:

```
<server xmlns="urn:jboss:domain:4.1">
        <extensions>
          ...list of extensions here
        </extensions>

        <system-
                properties> ...system properties defined here
        </system-properties>

        <management>
          ...management interfaces defined here </
        management>

        <profile> ...list
          of subsystems and their configurations </profile>


        <interfaces>
          ...interface definitions
        </interfaces>

        <socket-binding-group> ...socket
          binding definitions </socket-binding-
        group>


        <deployments> ...deployed applications go here
        </deployments> </
server>
```

extensions

Extensions are modules that extend the core capabilities of the server. A module is a package made up of a library developed in Java and an XML configuration file that offers Java EE compliant functionality.

An extension defines one or more subsystems based on a module. Included in the <extensions> element is a list of <extension> elements that make a subsystem available to that server. For example, the following excerpt describes a list of

extensions that are made available to a server, including ejb3, which is responsible for providing transactional functions, as required by the JEE 7 specifications:

```
    <extensions>
        <!-- list all extensions that you want made available to this server --> <extension
        module="org.jboss.as.clustering.infinispan"/> <extension
        module="org.jboss.as.deployment-scanner"/> <extension
        module="org.jboss.as.ejb3"/> <extension
        module="org.jboss.as.jpa"/> </extensions>
```

Each extension must be declared in the standalone.xml file and each module must be stored in JBOSS_HOME/modules/system/layers/base/. To manage the extension, EAP requires a <subsystem> tag where all the necessary customization for that subsystem will be declared. This topic will be discussed later during this course.

use

To find a module, EAP uses a module name, such as org.jboss.as.jpa, and finds the folder in the modules org/jboss/as/jpa/ directory to locate the module.

To demonstrate the <extensions> section of a configuration file, imagine an EAP instance that includes all the extensions for Java EE 7 Full Profile, as well as clustering and high availability as needed. The <extensions> element must, in this case, contain the following:

```
<extensions>
        <extension module="org.jboss.as.clustering.infinispan"/> <extension
        module="org.jboss.as.connector"/> <extension
        module="org.jboss.as.deployment-scanner"/> <extension
        module="org.jboss.as.ee"/> <extension
        module="org.jboss.as.ejb3"/> <extension
        module="org.jboss.as.jaxrs"/> <extension
        module="org.jboss.as.jdr"/> <extension
        module="org.jboss.as.jmx"/> <extension
        module="org.jboss.as.jpa"/> <extension
        module="org.jboss.as.jsf"/> <extension
        module="org.jboss.as.logging"/> <extension
        module="org.jboss.as.mail"/> <extension
        module="org.jboss.as.naming"/> <extension
        module="org.jboss.as.pojo"/> <extension
        module="org.jboss.as.remoting"/> <extension
        module="org.jboss.as.sar"/> <extension
        module="org.jboss.as.security"/> <extension
        module="org.jboss.as.transactions"/> <extension
        module="org.jboss.as.webservices"/> <extension
        module="org.jboss.as.weld"/> <extension
        module="org.wildfly.extension.batch.jberet"/> <extension
        module="org.wildfly.extension.bean-validation"/> <extension
        module="org.wildfly.extension.io"/> <extension
        module="org.wildfly.extension.request-controller"/> <extension
        module="org.wildfly.extension.security.manager"/> <extension
        module="org.wildfly.extension.undertow"/>
    </extensions>
```

Management interfaces

Following the <extensions> section in standalone.xml is the <management> section, which is used to define *management interfaces.* Management interfaces allow remote clients to connect to the EAP instance to manage it. EAP exposes two management interfaces:

- **HTTP Interface: Grants access to the administration console.**

- **Native Interface – Allows administrative tasks to be performed in compliance with a binary, proprietary protocol, which is what the Command Line Interface (CLI) tool uses.**

The <management> element contains the values necessary to make these interfaces available. The configuration looks as follows:

```
<management>
   ...
   <management-interfaces>
                  <native-interface security-realm="ManagementRealm"> <socket-
                        binding native="management-native"/> </native-interface>
                  <http-interface security-
                  realm="ManagementRealm"> <socket-binding http="management-
                        http"/> </http-interface> </management-interfaces>



   ...
</management>
```

The management-native and management-http socket bindings are defined later in the configuration file in the <socket-binding-group> section. This is where you define the actual host and port that these interfaces listen on.

use

A user can restrict access via the HTTP interface simply by removing it from the management-interfaces list:

```
<management>
       ...
      <management-interfaces>
             <native-interface security-realm="ManagementRealm"> <socket-
                     binding native="management-native"/> </native-interface> </
             management-interfaces>

 ...
</management>
```

This is the easiest way to disable the admin console (without an interface there will be no way to access it).

In the following XML, there is a <security-realm> tag called ManagementRealm and it is used by both management interfaces in the preceding XML. It refers to a text file called mgmt-users.properties, where the user's credentials are stored.

use

In a production environment, you can replace this domain with a database or LDAP domain. Security domains are discussed in detail later in this course.

Client applications that need remote access to EAP use the security-realm tag named ApplicationRealm .

```
<management>
 ...
   <security-realms>
                  <security-realm name="ManagementRealm">
                         <authentication>
                                <local default-user="$local" skip-group-loading="true"/> <properties path="mgmt-
                                users.properties" relative
to="jboss.server.config.dir"/>
                         </authentication>
                         <authorization map-groups-to-roles="false">
                                <properties path="mgmt-groups.properties" relative
to="jboss.server.config.dir"/>
                         </authorization>
                  </security-realm>
                  <security-realm name="ApplicationRealm">
                         <authentication> <local
                                default-user="$local" allowed-users="*" skip-group
loading="true"/>
                                <properties path="application-users.properties" relative
to="jboss.server.config.dir"/>
                         </authentication>
                         <authorization>
                                <properties path="application-roles.properties" relative
to="jboss.server.config.dir"/>
                         </authorization> </
                  security-realm> </
   security-realms>
  ...
</management>
```

Profiles and subsystems

A *profile* is a group of subsystems. A *subsystem* is where the extensions of the standalone EAP instance can be configured. A subsystem is added to a profile with the following two objectives:

- **If a subsystem appears within a profile, the subsystem will be made available to the server using that profile.**
- **The subsystem allows you to configure the extension to meet the needs user specific.**

When starting a standalone EAP server, there is only one profile available.

use

There is a direct relationship between a subsystem and its extension. A subsystem can only be added to a profile if the corresponding <extension> element appears in the <extensions> section.

The <profile> element has a group of child <subsystem> elements, with each <subsystem> entry consisting of the unique configuration values of that particular extension. Some subsystem definitions do not require a value, such as those for the jsf and jaxrs subsystems:

```
<profile>
     <subsystem xmlns="urn:jboss:domain:jsf:1.0"/> <subsystem
     xmlns="urn:jboss:domain:jaxrs:1.0"/>

     ...other subsystem definitions </profile>
```

use

jaxrs and jsf are configured in locations outside of subsystems. For example, jaxrs uses annotations and configurations are done within the source code. jsf uses its own configuration file, faces-config.xml.

Some subsystems have many configuration values, such as the datasources subsystem:

```
<profile>
    <subsystem xmlns="urn:jboss:domain:datasources:4.0"> <datasources>

            <datasource jndi-name="java:jboss/datasources/ExampleDS" pool
name="ExampleDS" enabled="true" use-java-context="true">
                <connection
url>jdbc:h2:mem:test;DB_CLOSE_DELAY=-1;DB_CLOSE_ON_EXIT=FALSE</connection-url>
                <driver>h2</driver>
                <security>
                    <user-name>to</user-name>
                    <password>to</password>
                </security> </
            datasource>
            <drivers>
                <driver name="h2" module="com.h2database.h2">
                    <xa-datasource-class>org.h2.jdbcx.JdbcDataSource</xa-datasource
class>
                </driver>
            </drivers> </
        datasources>
    </subsystem>
</profile>
```

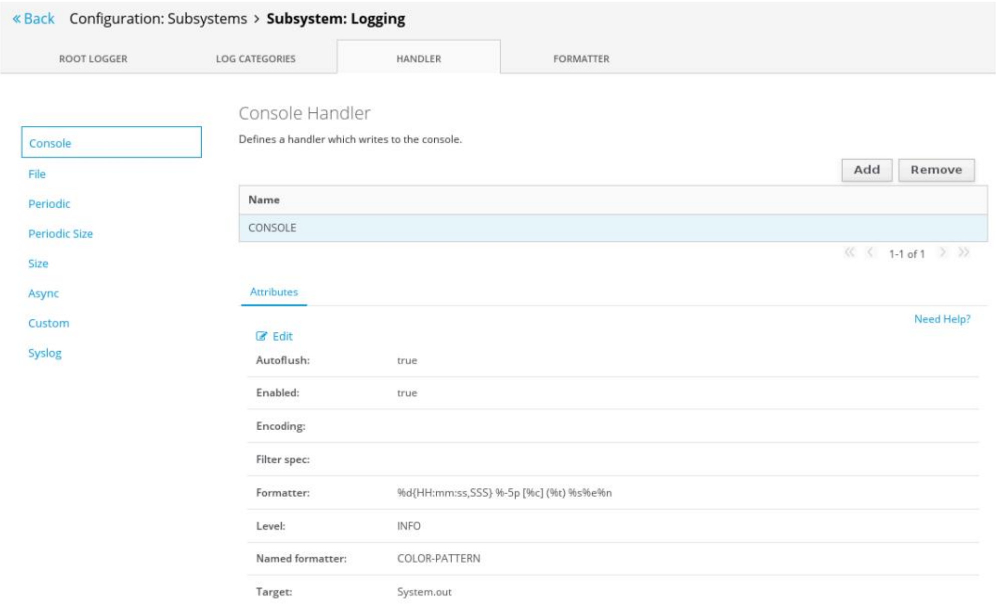
use

Child elements of <subsystem> are unique to each individual subsystem. See the corresponding schematic to learn and determine how to configure a particular subsystem. You will find the documents with schemas in the JBOSS_HOME/docs/schema folder.

For example, if you want to see the values for the data sources subsystem configuration, view the contents of its schema at JBOSS_HOME/docs/schema/wildfly-datasources_4_0.xsd.

Demo: Configuring profiles

We have seen the XML that represents the profiles and subsystems of a standalone server. It can be edited directly in standalone.xml. However, most common administrative tasks can and should be done with the management console. For example, the following screenshot shows how to edit log handlers in the admin console:



Console Handler

Defines a handler which writes to the console.

Name
CONSOLE

Autoflush:	true
Enabled:	true
Encoding:	
Filter spec:	
Formatter:	%d{HH:mm:ss,SSS} %-5p [%c] (%t) %s%e%n
Level:	INFO
Named formatter:	COLOR-PATTERN
Target:	System.out

Figure 2.2: Editing the log handler in the administration console

Watch the video to later perform the steps that must be followed. Play it again as many times as you deem necessary.

1. Open a terminal window from the workstation virtual machine (Applications > Favorites > Terminal) and run the following command to verify that EAP has been installed and is not currently running, and to create a new base directory for EAP:

```
[student@workstation ~]$ demo configuring-profiles setup
```

2. This demo will run a standalone EAP instance, using a custom server folder located at / home/student/JB248/labs/configuring profiles. Open the standalone.xml file and see the three <logger> entries in the logging subsystem. The file is located at /home/student/ JB248/labs/ configuring-profiles/configuration/standalone.xml.

```
 ...
<profile>
        <subsystem xmlns="urn:jboss:domain:logging:3.0">
                 ...
                <logger category="com.arjuna">
                        <level name="WARN"/>
                </logger>
                <logger category="org.jboss.as.config"> <level
                        name="DEBUG"/> </logger>
                <logger
                category="sun.rmi"> <level
                        name="WARN"/> </logger>

                 ...
        </subsystem>
 ...
<profile>
 ...
```

use

The Registry Subsystem will be covered in much more detail later in this course. For now, this demo will illustrate how the management console updates the standalone.xml file.

Ignore the following logger that appears outside of the profile, as it is outside the scope of this demo:

```
<logger log-boot="true" log-read-only="false" enabled="false"> <handlers> <handler
                                name="file"/>

                                </handlers>
</logger>
```

3. **Start the EAP server using the standalone.sh script in /opt/jboss-eap-7.0/ bin and specify the lab directory in /home/student/JB248/labs/ configuring-profiles/configuration/ standalone.xml as the directory server database. To access the EAP administration console to create a new logger, the standalone server must be running:**

```
[student@workstation ~]$ cd /opt/jboss-eap-7.0/bin
[student@workstation bin]$ ./standalone.sh \
-Djboss.server.base.dir=/home/student/JB248/labs/configuring-profiles/
```

Wait for the server to start completely.

4. Access the management console by visiting http://localhost:9990

use

The administrator username is jbossadm and the password is JBoss@RedHat123.

5. Click Settings at the top of the management console. Make Click Subsystems in the first column, and then click Records in the second column. Then click View to access the log subsystem configuration page.

6. Using the management console, add a new log category to the setting. Select RECORD CATEGORIES at the top of this page and click Add above the category list.

7. Notice that the currently populated categories are the same as what the loggers observed in the previous configuration file. Define the Category Name as org.jboss.as.clustering and set the level to DEBUG. Select Save. You should see the notification that the category was successfully saved.

8. Open the /home/student/JB248/labs/configuring-profiles/ configuration/standalone.xml file again and view the logging subsystem for changes reflected in the configuration file. You should see a fourth entry that looks like this:

```
 ...
<logger category="org.jboss.as.clustering" use-parent-handlers="false">
        <level name="DEBUG"/>
</logger>
 ...
```

9. Return to the administration console and remove the org.jboss.as.clustering category by clicking org.jboss.as.clustering and then clicking Remove. Confirm the deletion in the pop-up window to return the server to its original state.

10. Reopen the file /home/student/JB248/labs/configuring-profiles/ configuration/standalone.xml and notice that the number of loggers has returned to three in the logging subsystem.

11. Stop the EAP instance by pressing Ctrl+C in the terminal window that is running EAP.

This concludes the demo.

Quiz: Profile Settings

Choose the correct answer to the following questions:

1. Which sections in standalone.xml are required to add and configure a module on the EAP server? (Select two).

a. **Profile**

b. **extensions**

c. **administration interface**

d. **socket binding group**

2. Which section in standalone.xml is responsible for describing which users can access the admin console? (Select one).

a. **Profile**

b. **extensions**

c. **administration interface**

d. **System properties**

3. Which management interface is responsible for providing access to the management console? (Select one).

a. **native interface**

b. **HTTP interface**

c. **HTTPS interface**

d. **unsecured interface**

4. How many profiles can be defined in standalone.xml? (Choose one).

to. One b.

Four

c. **As many as needed.**

d. **Zero**

5. What methods are recommended to configure EAP? (Select two).

a. **Edit standalone.xml directly with a text editor.**

b. **The Management Console The EAP**

c. **Command Line Interface Make minor changes to**

d. **standalone.xml and major changes to the Management Console.**

Solution

Choose the correct answer to the following questions:

1. Which sections in standalone.xml are required to add and configure a module on the EAP server? (Select two).

a. **Profile**
b. **extensions**
c. administration interface
d. socket binding group

2. Which section in standalone.xml is responsible for describing which users can access the admin console? (Select one).

a. Profile
b. extensions
c. **administration interface**
d. System properties

3. Which management interface is responsible for providing access to the management console? (Select one).

a. native interface
b. **HTTP interface**
c. HTTPS interface
d. unsecured interface

4. How many profiles can be defined in standalone.xml? (Choose one).

to. One b.
Four
c. As many as needed.
d. Zero

5. What methods are recommended to configure EAP? (Select two).

a. Edit standalone.xml directly with a text editor.
b. **The Management Console The EAP**
c. **Command Line Interface** Make minor changes to
d. standalone.xml and major changes to the Management Console.

Configuring socket bonding groups and interfaces

Goals

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

• Configure JBoss EAP network interfaces and socket binding groups.

Interfaces

An interface is a logical name given to a network interface, IP address, or hostname to which a socket can be attached. The <interfaces> element is used to define an interface by using the <interface> child element. For example, the <interfaces> section defined in the default standalone.xml configuration file looks like this:

```
<interfaces>
        <interface name="management">
            <inet-address value="${jboss.bind.address.management:127.0.0.1}"/>
        </interface>
        <interface name="public"> <inet-
            address value="${jboss.bind.address:127.0.0.1}"/>
        </interface>
</interfaces>
```

use

The properties in $ and braces are system runtime properties. The value after the colon is the default if the runtime property is not defined.

There are three predefined interfaces, but users can add as many as are needed, depending on the user's particular network and hardware. The default value for each interface is 127.0.0.1, which represents the loopback network interface (localhost).
By binding to localhost, an EAP immediate installation will not be exposed to the outside as a security feature. The interface named "admin" is kept separate from the interface named "public" to facilitate secure access to the EAP management features.

Users can configure the IP address to access the management console, when starting EAP, using the -bmanagement command line variable. For example:

```
$ ./standalone.sh -bmanagement 127.0.0.1
```

The purpose of the <interfaces> element is to provide a single location for defining physical addresses used in an environment; you can then reference those physical addresses as many times as necessary throughout the configuration file. Do not specify ports or protocols here. These details are defined later in socket binding groups.

The interface can have any name value and there are several options for defining interfaces. If a host has multiple network interface cards (NICs) with multiple IP addresses and only one NIC needs to be exposed, use the first of the following interface configurations. If all network interfaces must be exposed, use the second interface configuration.

```
<interfaces>
        <interface name="my_ip_address">
                <inet-address value="128.164.0.15"/>
        </interface> </
interfaces>
```

The following interface uses the any-address wildcard to bind to all addresses IP available on the server:

```
<interfaces>
        <interface name="global"> <any-
                address/>
        </interface> </
interfaces>
```

The above interface demonstrates the use of an *interface criteria,* which is used at runtime to determine the IP address to use for an interface. The following interface also uses a criteria that will only bind to an IPv4 address:

```
<interfaces>
        <interface name="ipv4-global"> <any-
                ipv4-address/> </interface>
        </interfaces>
```

Criteria can be combined to be highly specific in defining an interface. For example, the following interface defines a specific subnet that supports multicast. The address must be active and cannot be limited to two extremes:

```
<interfaces>
        <interface name="my_specific_interface">
                <subnet-match value="192.168.0.0/16"/>
                <up/>
                <multicast/>
                <not>
                        <point-to-point/> </not>

        </interface> </
interfaces>
```

In a Linux™ environment, users can define an interface for a specific network card:

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

Socket binding groups

A *socket binding group* is a set of named socket bindings, which allows users to define all the ports required for an EAP instance. A socket binding group is defined by the <socket-binding-group> element, which consists of a set of <socket-binding> child elements. For example, here is the default socket binding group defined in standalone.xml:

```
<socket-binding-group name="standard-sockets" default-interface="public" port  offset="${jboss.socket.binding.port-
offset:0}"> <socket-binding name="management-http"
                interface="management" port="${jboss.management.http.port:9990}"/> <socket-binding
  name="management-https" interface="management" port="$
                {jboss.management.https.port:9993}"/> <socket-binding name="ajp" port="$
  {jboss.ajp.port:8009}"/> <socket-binding name="http" port="$
                {jboss.http.port:8080}"/> <socket-binding name="https" port="${jboss.https.port:8443}"/
                > <socket-binding name="txn-recovery-environment" port="4712"/> <socket-binding
                name="txn-status-manager" port="4713"/> <outbound-socket-binding name="mail-smtp">



                        <remote-destination host="localhost" port="25"/>
                </outbound-socket-binding>
</socket-binding-group>
```

The name of this socket binding group is standard-sockets, but any name can be used when defining a socket binding group. The default-interface attribute must be set to a named interface defined in the <interfaces> section. Each <socket-binding> entry has a protocol name and a port number. Note that <socket-binding> can also define an interface attribute indicating an <interface> definition, which overrides the default interface value.

port-offset allows users to easily modify all port numbers by specifying a positive offset value. This is particularly useful in situations where multiple EAP instances are running on the same host and an offset can be used to avoid port conflicts that prevent an instance from starting. Note that port-offset can be set via the jboss.socket.binding.port-offset runtime property. This allows users to provide the port offset for the server as a property when starting the server.

A new feature in EAP 7 is the port reduction facility. EAP port drop tunneling protocols over HTTP using the HTTP UPGRADE method. It allows different protocols to use the same port, such as those provided by EJBs, messaging, or any other subsystem that can be accessed externally.

Quiz: Configuring Socket Binding Groups and Interfaces

Based on the settings provided, select the correct answers to the following questions:

```
  <interfaces>
              <interface name="management"> <inet-
                     address value="${jboss.bind.address.management:192.168.0.1}"/>
              </interface>
              <interface name="public"> <inet-
                     address value="${jboss.bind.address:127.0.0.1}"/>
              </interface>
              <interface name="global"> <any-
                   address/> </
              interface>
       </interfaces>
       <socket-binding-group name="standard-sockets" default-interface="public" port
offset="${jboss.socket.binding.port-offset:10000}"> <socket-binding
              name="management-http" interface="management" port="${jboss.management.http.port:9990}"/
  >
              <socket-binding name="management-https" interface="management" port="$
  {jboss.management.https.port:9993}"/> <socket-binding
              name="ajp" port="${jboss.ajp.port:8009}"/> <socket-binding name="http" port="$
              {jboss.http.port:8080}"/> <socket-binding name="https" port="${jboss.https.port:8443}"/>
              <socket-binding name="txn-recovery-environment" port="4712"/> <socket-binding name="txn-
              status-manager" port="4713"/> <outbound-socket-binding name="mail-smtp">



                     <remote-destination host="localhost" port="25"/>
              </outbound-socket-binding> </socket-
       binding-group>
```

1. Which of the following commands does NOT start a standalone EAP server that exposes the public IP address to 127.0.0.1, based on the configuration? (Choose one).

a. **Run from a terminal window:**

```
$ standalone.sh -Djboss.bind.address.management=0.0.0.0 \
 -Djboss.bind.address=205.25.238.172
```

b. **Run from a terminal window:**

```
$ standalone.sh -Djboss.bind.address=127.0.0.1
```

c. **Run from a terminal window:**

```
$ standalone.sh
```

d. **Run from a terminal window:**

```
$ standalone.sh -Djboss.bind.address=0.0.0.0
```

2. What port is used by the EAP management console, based on the previous configuration with all defaults used? (Choose one).

a. 9990 b. 8080

19990 18080

c.

d.

3. What interfaces are bound to 127.0.0.1 in the above configuration? (Select a).

a. Global

b. Public

c.

d. Management None of the above.

4. What ports can be used with HTTPS binding, depending on the configuration former? (Choose one).

a. 8443 b.

8443, 9993

c. 18443

d. 18443, 19993

Solution

Based on the settings provided, select the correct answers to the following questions:

```
  <interfaces>
              <interface name="management"> <inet-
                     address value="${jboss.bind.address.management:192.168.0.1}"/>
              </interface>
              <interface name="public"> <inet-
                     address value="${jboss.bind.address:127.0.0.1}"/>
              </interface>
              <interface name="global"> <any-
                   address/> </
              interface> </
       interfaces> <socket-
       binding-group name="standard-sockets" default-interface="public" port
offset="${jboss.socket.binding.port-offset:10000}"> <socket-binding
              name="management-http" interface="management" port="${jboss.management.http.port:9990}"/
  >
              <socket-binding name="management-https" interface="management" port="$
  {jboss.management.https.port:9993}"/> <socket-binding
              name="ajp" port="${jboss.ajp.port:8009}"/> <socket-binding name="http" port="$
              {jboss.http.port:8080}"/> <socket-binding name="https" port="${jboss.https.port:8443}"/>
              <socket-binding name="txn-recovery-environment" port="4712"/> <socket-binding name="txn-
              status-manager" port="4713"/> <outbound-socket-binding name="mail-smtp">


                     <remote-destination host="localhost" port="25"/>
              </outbound-socket-binding> </socket-
       binding-group>
```

1. Which of the following commands does NOT start a standalone EAP server that exposes the public IP address to 127.0.0.1, based on the configuration? (Choose one).

a. **Run from a terminal window:**

```
$ standalone.sh -Djboss.bind.address.management=0.0.0.0 \
 -Djboss.bind.address=205.25.238.172
```

b. **Run from a terminal window:**

```
$ standalone.sh -Djboss.bind.address=127.0.0.1
```

c. **Run from a terminal window:**

```
$ standalone.sh
```

d. **Run from a terminal window:**

```
$ standalone.sh -Djboss.bind.address=0.0.0.0
```

2. What port is used by the EAP management console, based on the previous configuration with all defaults used? (Choose one).

a. 9990 b. 8080

c. 19990

d. 18080

3. What interfaces are bound to 127.0.0.1 in the above configuration? (Select a).

a. Global

b. Public

c.

d. Management None of the above.

4. What ports can be used with HTTPS binding, depending on the configuration former? (Choose one).

a. 8443 b.

8443, 9993

c. 18443

d. 18443, 19993