Guided Exercise: Control Log Levels

In this lab, you will manipulate the file handler logging levels that rotate based on size that you created in the previous lab, and implement an application that uses this handler to log messages.

Resources	
Files:	/home/student/JB248/labs/standalone /home/student/JB248/labs/logging-levels
App URL:	http://localhost:8080/logtest
Resources	/home/student/JB248/labs/logging-levels/ logtest.war

Results

You should be able to change the file handler log levels that rotate based on size and see application-generated log messages in the EAP server log files.

before you start

Before beginning the guided exercise, run the following command to verify that EAP has been installed to /opt/jboss-eap-7.0, that no EAP instances are running, that the previous guided exercise has completed, and that a handler is defined of files that rotate based on size, as well as downloading the logtest.war application:

```
[student@workstation ~]$ lab logging-levels setup
```

1. Start the standalone EAP server.

Use the following command to start an EAP instance using the /home/student/JB248/labs/standalone folder as the base directory:

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

Before proceeding, wait for the server to finish starting up.

2. Verify that a handler for files that rotate based on the file has been defined. size named FILE_BY_SIZE_ROTATING using the EAP CLI. Open a new terminal window and run the following commands:

```
[student@workstation ~]$ cd /opt/jboss-eap-7.0/bin
[student@workstation bin] ./jboss-cli.sh --connect
[standalone@localhost:9990] /subsystem=logging/\ size-rotating-
file-handler=FILE_BY_SIZE_ROTATING:read-resource
```

The result should appear as follows:

```
{
    "outcome" => "success",
    "result" =>
        { "append" => true,
        "autoflush" => true,
        "enabled" => true,
        "encoding" => undefined, "file"
        => {
            "relative-to" => "jboss.server.log.dir", "path" =>
            "production-server.log"
        },
        "filter" => undefined, "filter-
        spec" => undefined, "formatter" =>
        "%d{HH:mm:ss,SSS} %-5p [%c] (%t) %s%E%n", "level" => "INFO", "max-
        backup-index" => 3,
        "name" =>
        "FILE_BY_SIZE_ROTATING", "named-
        formatter" => undefined, "rotate-on-boot"
        => false, "rotate-size" => "1024m",
        "suffix" => undefined

    }
}
```

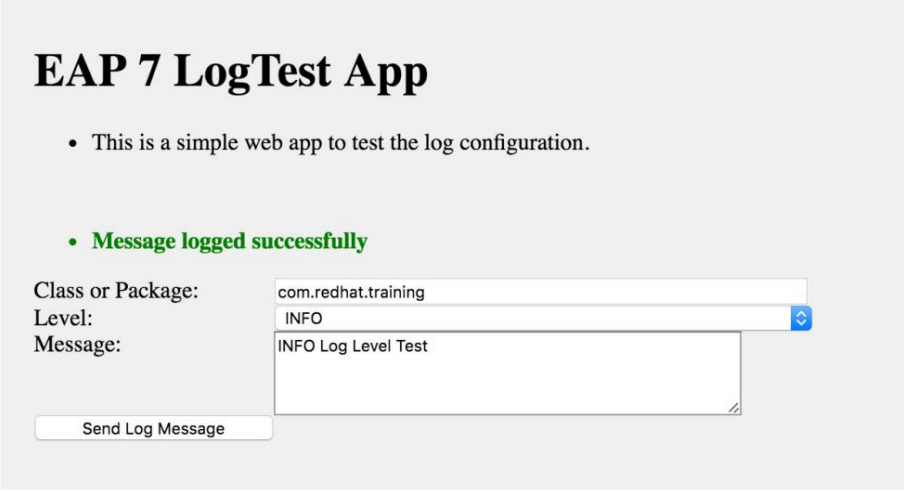
3. You will now implement the logtest.war file in EAP to generate messages in different levels of registration.

Run the following command:

```
[standalone@localhost:9990] deploy \ /home/
student/JB248/labs/logging-levels/logtest.war
```

4. Test the log levels.

4.1. Navigate to http://127.0.0.1:8080/logtest/ to access the application logtest.



4.2. • Enter com.redhat.training.view in the Class or package field.

• Select the INFO level from the Level dropdown list.

• Enter the text INFO Log Level Test in the Message field.

Click Send Log Messages to send the log message to the handler you created in the previous step.

4.3. Open a new terminal window to view the log file in /home/student/JB248/labs/standalone/log/production-server.log and verify that you can see the messages logged by the application.

```
[student@workstation ~]$ tail -f \ /home/student/
JB248/labs/standalone/log/production-server.log
...
04:09:53,623 INFO [com.redhat.training.view] (default task-27) INFO Log Level
  Test
```

4.4. Send a few more Application INFO messages and verify that the messages appear in the log file.

4.5. Change the Level field to DEBUG in the logtest application and verify that the messages do NOT appear in the log file. This is because the handler is configured to only process INFO and all levels below INFO in the logger hierarchy, while DEBUG is higher than INFO in the logger hierarchy.

4.6. Change the Level field to WARN in the logtest application and verify that the messages appear in the log file. This is because the handler is configured to process INFO and all levels below INFO in the logger hierarchy.

4.7. Now modify the default log level of the handler FILE_BY_SIZE_ROTATING a DEBUG.

Run the following command:

```
[standalone@localhost:9990] /subsystem=logging/\ size-rotating-
file-handler=FILE_BY_SIZE_ROTATING:\ write-
attribute(name=level,value=DEBUG)
```

4.8. Change the Level field to DEBUG in the logtest application and verify that the messages now appear in the log file. This is because the handler is now configured to process DEBUG and all levels below DEBUG in the logger hierarchy.

4.9. Change the Level field to INFO in the logtest application and verify that the messages appear in the log file. This is because INFO is below DEBUG in the logger hierarchy.

4.10. Change the Level field to TRACE in the logtest application and verify that the messages do NOT appear in the log file. This is because TRACE is higher than DEBUG in the logger hierarchy and the handler will not process messages at this level.

5. Perform cleaning.

5.1. Undeploy the logtest.war application:

```
[standalone@localhost:9990 /] undeploy logtest.war
```

5.2. Exit the EAP CLI:

```
[standalone@localhost:9990 /] exit
```

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

Stop the tail command by pressing Ctrl+C in the terminal window that is running the tail command.

This concludes the guided exercise.

Lab Work: Configuring the Logging Subsystem

In this lab work, you will configure the logging subsystem in an EAP managed domain and implement the bookstore application to verify logging.

Resources	
Files	/opt/domain /var/log/jboss
app url	http://172.25.250.10:8080/bookstore http:// 172.25.250.11:8080/bookstore

Result

You must be able to configure the registration subsystem of an EAP 7 managed domain and implement the bookstore application to verify that application registrations are generated in a single, centralized location.

before you start

Use the following command to download the relevant lab files and ensure that the managed domain is set correctly:

```
[student@workstation ~]$ lab logging-lab-final setup
```

1. You can use the EAP 7 Management Console or the JBoss EAP CLI to achieve your goals, although the EAP CLI is the preferred option in production environments.

An EAP administrator has configured a managed domain with two host controllers running the servera and serverb virtual machines, respectively, and the domain controller on the workstation. The domain and host configuration files are stored in the /opt/domain folder on all three machines. You must start the managed domain and configure the logging subsystem of the full-ha profile in the managed domain.

As a final step, you should deploy the bookstore WAR file to the managed domain and verify that the application logs for all server instances in the managed domain were generated to the /var/log/jboss folder from an NFS mount point. already configured and pointing to the workstation virtual machine.

2. Start the domain controller on workstation. Because the files domain controller configuration are kept in the /opt/domain folder on workstation, use /opt/domain as the value of the jboss.domain.base.dir argument that you pass to the domain.sh startup script. Also notice that the host file for the domain controller is named host-master.xml and is located in the /opt/domain/configuration folder. (Tip: send the --host-config=host master.xml argument to domain.sh.)

Note that the /opt/domain directory is owned by the jboss user, so you must start the domain controller using sudo -u jboss /opt/jboss-eap-7.0/ bin/domain.sh ...

3. The two host controllers on servera and serverb connect to the host controller domain in the previous step and get the latest configuration of the domain. Start the two host controllers on servera and serverb.

   3.1. Start the host controller on servera. Because the host controller configuration files are kept in the /opt/domain folder on servera, use /opt/domain as the value of the jboss.domain.base.dir argument that you pass to the domain.sh startup script. Also notice that the host file for the host controller is named host-slave.xml and is located in the /opt/domain/ configuration folder. (Hint: Pass the --host config=host-slave.xml argument to domain.sh.)

   Note that the /opt/domain directory is owned by the jboss user, so start the host controller using sudo -u jboss /opt/jboss eap-7.0/bin/domain.sh ...

   3.2. Start the host controller on serverb. Because the host controller configuration files are kept in the /opt/domain folder on serverb, use /opt/domain as the value of the jboss.domain.base.dir argument that you pass to the domain.sh startup script. Also notice that the host file for the host controller is named host-slave.xml and is located in the /opt/domain/ configuration folder. (Hint: Pass the --host config=host-slave.xml argument to domain.sh.)

   Note that the /opt/domain directory is owned by the jboss user, so start the host controller using sudo -u jboss /opt/jboss eap-7.0/bin/domain.sh ...

   3.3. Verify that both host controllers connect to the domain controller and form a managed domain. Look at the console window in which you started the domain controller and verify that both servera and serverb are registered as slaves to the domain controller.

4. By default, the logging subsystem is configured with two handlers: console-handler that logs messages to the console; and periodic rotating-file-handler that logs messages to a file in the /opt/domain/servers/ <SERVER_NAME>/log folder on servera and serverb. These log files are rotated daily.

   To easily back up and archive the EAP log files, your system administrator has asked you to set up a size-based rotating file handler that stores all server logs in a single, domain-wide, central location. managed. This central logging location is the /var/log/jboss folder on the servera and serverb virtual machines.
   This folder is configured as a remote NFS share mounted from the workstation virtual machine. Because the names of the server instances in the managed domain are unique, the logs for a particular instance must be stored in the appropriately named folder. For example, if a server

is named server-one, the logs should be stored in the /var/log/jboss/server-one/ folder.

Create a new size-rotating-file-handler to address this requirement. You can use the EAP management console or the EAP CLI to achieve your goal.

4.1. Recall from the demo and the guided exercises that the size-rotating-file handler accepts a relative-path value and a filename as input. Generally, relative paths indicate the log directory in the EAP base directory. In this lab, you need to change the relative-path value to point to the /var/log/ jboss folder.

We may change the centralized log folder path in the future, so you should avoid referencing the full path, and instead use the EAP concept of a path variable . The *path* variable refers to the logical name of a file system path. The domain.xml, host.xml, and standalone.xml configurations include a section where routes must be declared. Other sections of the configuration can then refer to those routes by their logical names, instead of having to include full route details.

Access the JBoss EAP CLI. In a new terminal window on workstation, start the EAP CLI and connect to the domain controller as the jboss user:

Create a new *path* variable called custom.log.dir and set its value to /var/log/ jboss. In the EAP management console, this can be done in the Configuration > Routes section.

4.2. After you have defined a new route variable, you can reference it when creating handlers. Create a new size-rotating-file-handler named BOOKSTORE_LOG_HANDLER with the following characteristics:

- **Nombre: BOOKSTORE_LOG_HANDLER**
- **Path (log file name): ${jboss.server.name}/ bookstore.log**
- **Relative to (relative route variable): custom.log.dir**
- **Attach, Auto Dump, Enabled: Enable all of these attributes (if using the CLI, set them to 'true')**
- **Nivel: DEBUG**
- **Rotation size: 1m (1 MB)**
- **Maximum backup index: 5**

4.3. Verify the new handler that you created in the previous step using the JBoss EAP CLI or management console.

5. In this lab work, you should only implement the bookstore application in the managed domain. You've already set up a new handler to capture the application's log files. In production environments, the recommended approach is to set the ROOT logger to a very low level, such as ERROR or WARN, to avoid a

too verbose log output, and to set application-specific logger categories at the INFO or DEBUG level to collect and analyze application-specific problems.

Create a new logger category named com.redhat.training and set it to the DEBUG level by default, which is the package hierarchy for the bookstore application. Attach this logger to the BOOKSTORE_LOG_HANDLER you created in the previous step so that bookstore-related log messages appear in the bookstore.log file.

Change the logger level ROOT to WARN.

6. Reload the configuration of all the servers in the domain to apply the registry changes made so far.

6.1. Reload the configuration of all servers in the Group1 server group.

Verify that bookstore.war is implemented on servera.1 and serverb.1 as they are part of the Group1 server group. Watch the servera and serverb console window for any error messages or warnings. Because you reset the ROOT logger to the WARN level, you should see only the WARN and ERROR level messages in the servera and serverb console window.

6.2. Verify that you can access the bookstore application at http://172.25.250.10:8080/bookstore y http://172.25.250.11:8080/ bookstore.

6.3. Verify that you can see the bookstore application logs in the / folders var/log/jboss/servera.1 and /var/log/jboss/serverb.1. Notice how each server instance has its own folder under which the log files are created. Once the log file reaches 1 MB (rotation size), additional log files with a numbered suffix are created and rotated until the number of files reaches the max-backup-index value.

You should see the following DEBUG level logs in the bookstore.log for servera.1 and serverb.1:

```
15:06:18,563 DEBUG [com.redhat.training.utils.DatabasePopulator] (ServerService Thread Pool -- 90)
  Loaded catalog successfully!!!
15:06:18,647 DEBUG [com.redhat.training.utils.DatabasePopulator] (ServerService Thread Pool -- 90)
  Loaded Customers successfully!!!
15:06:18,649 DEBUG [com.redhat.training.utils.DatabasePopulator] (ServerService Thread Pool -- 90)
  Loaded Promotions successfully!!!
```

7. Using the JBoss EAP administrative console or CLI, undeploy the bookstore application and power off the servers, pools, host controllers, and the entire managed domain.

7.1. Undeploy the bookstore application.

7.2. Verify that the bookstore app is no longer accessible from servera.1 and serverb.1.

7.3. Stop all servers in Group1. Verify that the bookstore app is no longer be accessible.

7.4. Close the host controller on servera. Observe the servera console window and verify that the host controller has been shut down.

7.5. Close the host controller in serverb. Look at the serverb console window and verify that the host controller has been shut down.

8. Perform cleaning and grading.

8.1. Press Ctrl+C to stop the domain controllers. Alternatively, you can shut down the domain controller using the JBoss EAP CLI command / host=master:shutdown()).

8.2. Run the following workstation command to grade the assignment:

```
[student@workstation bin]$ lab logging-lab-final grade
```

This concludes the lab work.

Solution

In this lab work, you will configure the logging subsystem in an EAP managed domain and implement the bookstore application to verify logging.

Resources	
Files	/opt/domain /var/log/jboss
app url	http://172.25.250.10:8080/bookstore http://172.25.250.11:8080/bookstore

Result

You must be able to configure the registration subsystem of an EAP 7 managed domain and implement the bookstore application to verify that application registrations are generated in a single, centralized location.

before you start

Use the following command to download the relevant lab files and ensure that the managed domain is set correctly:

```
[student@workstation ~]$ lab logging-lab-final setup
```

1. You can use the EAP 7 Management Console or the JBoss EAP CLI to achieve your goals, although the EAP CLI is the preferred option in production environments.

An EAP administrator has configured a managed domain with two host controllers running the servera and serverb virtual machines, respectively, and the domain controller on the workstation. The domain and host configuration files are stored in the /opt/domain folder on all three machines. You must start the managed domain and configure the logging subsystem of the full-ha profile in the managed domain.

As a final step, you should deploy the bookstore WAR file to the managed domain and verify that the application logs for all server instances in the managed domain were generated to the /var/log/jboss folder from an NFS mount point. already configured and pointing to the workstation virtual machine.

2. Start the domain controller on workstation. Because the files domain controller configuration are kept in the /opt/domain folder on workstation, use /opt/domain as the value of the jboss.domain.base.dir argument that you pass to the domain.sh startup script. Also notice that the host file for the domain controller is named host-master.xml and is located in the /opt/domain/configuration folder. (Tip: send the --host-config=host master.xml argument to domain.sh.)

Note that the /opt/domain directory is owned by the jboss user, so you must start the domain controller using sudo -u jboss /opt/jboss-eap-7.0/ bin/domain.sh ...

```
[student@workstation ~]$ sudo -u jboss /opt/jboss-eap-7.0/bin/domain.sh \
```

```
-Djboss.domain.base.dir=/opt/domain/ --host-config=host-master.xml
```

3. The two host controllers on servera and serverb connect to the host controller domain in the previous step and get the latest configuration of the domain. Start the two host controllers on servera and serverb.

   3.1. Start the host controller on servera. Because the host controller configuration files are kept in the /opt/domain folder on servera, use /opt/domain as the value of the jboss.domain.base.dir argument that you pass to the domain.sh startup script. Also notice that the host file for the host controller is named host-slave.xml and is located in the /opt/domain/ configuration folder. (Hint: Pass the --host config=host-slave.xml argument to domain.sh.)

   Note that the /opt/domain directory is owned by the jboss user, so start the host controller using sudo -u jboss /opt/jboss eap-7.0/bin/domain.sh ...

   Open a new terminal window on the server virtual machine and run the following command:

```
[student@servera ~]$ sudo -u jboss /opt/jboss-eap-7.0/bin/domain.sh \
-Djboss.domain.base.dir=/opt/domain/ \
-Djboss.domain.master.address=172.25.250.254 \ --host-
config=host-slave.xml
```

   3.2. Start the host controller on serverb. Because the host controller configuration files are kept in the /opt/domain folder on serverb, use /opt/domain as the value of the jboss.domain.base.dir argument that you pass to the domain.sh startup script. Also notice that the host file for the host controller is named host-slave.xml and is located in the /opt/domain/ configuration folder. (Hint: Pass the --host config=host-slave.xml argument to domain.sh.)

   Note that the /opt/domain directory is owned by the jboss user, so start the host controller using sudo -u jboss /opt/jboss eap-7.0/bin/domain.sh ...

   Open a new terminal window on the serverb virtual machine and run the following command:

```
[student@serverb ~]$ sudo -u jboss /opt/jboss-eap-7.0/bin/domain.sh \
-Djboss.domain.base.dir=/opt/domain/ \
-Djboss.domain.master.address=172.25.250.254 \ --host-
config=host-slave.xml
```

   3.3. Verify that both host controllers connect to the domain controller and form a managed domain. Look at the console window in which you started the domain controller and verify that both servera and serverb are registered as slaves to the domain controller.

4. By default, the logging subsystem is configured with two handlers: console-handler that logs messages to the console; and periodic-

rotating-file-handler that logs messages to a file in the /opt/domain/servers/<SERVER_NAME>/log folder on servera and serverb. These log files are rotated daily.

To easily back up and archive the EAP log files, your system administrator has asked you to set up a size-based rotating file handler that stores all server logs in a single, domain-wide, central location. managed. This central logging location is the /var/log/jboss folder on the servera and serverb virtual machines.
This folder is configured as a remote NFS share mounted from the workstation virtual machine. Because the names of the server instances in the managed domain are unique, the logs for a particular instance must be stored in the appropriately named folder. For example, if a server is named server-one, the logs should be stored in the /var/log/jboss/server-one/ folder.

Create a new size-rotating-file-handler to address this requirement. You can use the EAP management console or the EAP CLI to achieve your goal.

4.1. Recall from the demo and the guided exercises that the size-rotating-file handler accepts a relative-path value and a filename as input.
Generally, relative paths indicate the log directory in the EAP base directory. In this lab, you need to change the relative-path value to point to the /var/log/jboss folder.

We may change the centralized log folder path in the future, so you should avoid referencing the full path, and instead use the EAP concept of a path variable . The *path* variable refers to the logical name of a file system path. The domain.xml, host.xml, and standalone.xml configurations include a section where routes must be declared. Other sections of the configuration can then refer to those routes by their logical names, instead of having to include full route details.

Access the JBoss EAP CLI. In a new terminal window on workstation, start the EAP CLI and connect to the domain controller as the jboss user:

```
[student@workstation ~]$ sudo -u jboss \ /opt/jboss-
eap-7.0/bin/jboss-cli.sh \ --connect --
controller=172.25.250.254:9990
```

Create a new *path* variable called custom.log.dir and set its value to /var/log/jboss. In the EAP management console, this can be done in the Configuration > Routes section.

```
[domain@172.25.250.254:9990 /] /path=custom.log.dir:\ add(path=/var/
log/jboss/) {

        "outcome" => "success",
        ...
}
```

4.2. After you have defined a new route variable, you can reference it when creating handlers. Create a new size-rotating-file-handler called BOOKSTORE_LOG_HANDLER with the following characteristics:

- **Nombre: BOOKSTORE_LOG_HANDLER**
- **Path (log file name): ${jboss.server.name}/ bookstore.log**
- **Relative to (relative route variable): custom.log.dir**
- **Attach, Auto Dump, Enabled: Enable all of these attributes (if using the CLI, set them to 'true')**
- **Nivel: DEBUG**
- **Rotation size: 1m (1 MB)**
- **Maximum backup index: 5**

```
[domain@172.25.250.254:9990 /] /profile=full-ha/subsystem=logging/\ size-rotating-file-
handler=BOOKSTORE_LOG_HANDLER:add\ (file={"relative-
 to"=>"custom.log.dir",\ "path"=>"${jboss.server.name}/
 bookstore.log"}, \ enabled=true, append=true, autoflush=true,
\ rotate-size=1m, max-backup-index=5,\ level=DEBUG) {




        "outcome" => "success",
        ...
}
```

4.3. Verify the new handler that you created in the previous step using the JBoss EAP CLI or management console.

```
[domain@172.25.250.254:9990 /] /profile=full-ha/subsystem=logging\ /size-rotating-file-
handler=BOOKSTORE_LOG_HANDLER:read-resource {

        "outcome" => "success",
        ...
        }
}
```

5. In this lab work, you should only implement the bookstore application in the managed domain. You've already set up a new handler to capture the application's log files. In production environments, the recommended approach is to set the ROOT logger to a very low level, such as ERROR or WARN, to avoid too verbose logging output, and to set application-specific logger categories to the INFO or DEBUG level in order to to collect and analyze application-specific problems.

Create a new logger category named com.redhat.training and set it to the DEBUG level by default, which is the package hierarchy for the bookstore application. Attach this logger to the BOOKSTORE_LOG_HANDLER that you created in the

step above to display bookstore-related log messages in the bookstore.log file.

```
[domain@172.25.250.254:9990 /] /profile=full-ha/subsystem=logging/\
logger=com.redhat.training:add\
(category=com.redhat.training,level=DEBUG,handlers=["BOOKSTORE_LOG_HANDLER"]) {

        "outcome" => "success",
        ...
}
```

Change the logger level ROOT to WARN.

```
[domain@172.25.250.254:9990 /] /profile=full-ha/subsystem=logging/\ root-
logger=ROOT:write-attribute(name=level,value=WARN) {

        "outcome" => "success",
        ...
}
```

6. Reload the configuration of all the servers in the domain to apply the registry changes made so far.

6.1. Reload the configuration of all servers in the Group1 server group.

```
[domain@172.25.250.254:9990 /] /server-group=Group1:reload-servers
```

Verify that bookstore.war is implemented on servera.1 and serverb.1 as they are part of the Group1 server group. Watch the servera and serverb console window for any error messages or warnings. Because you reset the ROOT logger to the WARN level, you should see only the WARN and ERROR level messages in the servera and serverb console window.

6.2. Verify that you can access the bookstore application at http://172.25.250.10:8080/bookstore y http://172.25.250.11:8080/ bookstore.

6.3. Verify that you can see the bookstore application logs in the / folders var/log/jboss/servera.1 and /var/log/jboss/serverb.1. Notice how each server instance has its own folder under which the log files are created. Once the log file reaches 1 MB (rotation size), additional log files with a numbered suffix are created and rotated until the number of files reaches the max-backup-index value.

You should see the following DEBUG level logs in the bookstore.log for servera.1 and serverb.1:

```
15:06:18,563 DEBUG [com.redhat.training.utils.DatabasePopulator] (ServerService Thread Pool -- 90) Loaded
  catalog successfully!!!
15:06:18,647 DEBUG [com.redhat.training.utils.DatabasePopulator] (ServerService Thread Pool -- 90) Loaded
  Customers successfully!!!
```

```
15:06:18,649 DEBUG [com.redhat.training.utils.DatabasePopulator] (ServerService Thread Pool -- 90) Loaded
  Promotions successfully!!!
```

7. Using the JBoss EAP administrative console or CLI, undeploy the bookstore application and power off the servers, pools, host controllers, and the entire managed domain.

7.1. Undeploy the bookstore application.

```
[domain@172.25.250.254:9990 /] undeploy bookstore.war \ --all-relevant-
server-groups
```

7.2. Verify that the bookstore app is no longer accessible from servera.1 and serverb.1.

Verify that the bookstore application is NOT accessible at the URL http://172.25.250.10:8080/bookstore and http://172.25.250.11:8080/bookstore.

7.3. Stop all servers in Group1. Verify that the bookstore app is no longer be accessible.

```
[domain@172.25.250.254:9990 /] /server-group=Group1:\ stop-
servers(blocking=true) {

        "outcome" => "success",
        ...
}
```

7.4. Close the host controller on servera. Observe the servera console window and verify that the host controller has been shut down.

```
[domain@172.25.250.254:9990 /] /host=servera:shutdown() {

        "outcome" => "success",
        ...
}
```

7.5. Close the host controller in serverb. Look at the serverb console window and verify that the host controller has been shut down.

```
[domain@172.25.250.254:9990 /] /host=serverb:shutdown() {

        "outcome" => "success",
        ...
}
```

8. Perform cleaning and grading.

8.1. Press Ctrl+C to stop the domain controllers. Alternatively, you can shut down the domain controller using the JBoss EAP CLI command /host=master:shutdown()).

8.2. Run the following workstation command to grade the assignment:

```
[student@workstation bin]$ lab logging-lab-final grade
```

This concludes the lab work.

Summary

In this chapter, you learned the following:

• The EAP registration subsystem consists of three components:

ÿ Handlers: determine "where" and "how" an event will be logged. EAP comes with several built-in handlers that can be used to log application messages.

ÿ Loggers: Organize events into logically related categories. a category it is usually mapped to some package namespace that is running inside the JVM.

ÿ Root Logger—Loggers are organized in a parent-child hierarchy, with the root logger residing at the top of the logger hierarchy.

• EAP comes with a number of useful built-in handlers that can be used to configure the registry subsystem.

• By default, there are two handlers defined in EAP: a handler CONSOLE, which logs messages to the system console (terminal); and a FILE handler, which logs to a file named server.log. Both handlers parse log messages at the INFO level by default.

• Logger categories can be used for detailed control of which events to register. A logger can reference one or more handlers.

• Log levels are also organized in a hierarchical manner that determines the detail level of the log. Registration occurs at different levels and you can set the minimum level according to your specific needs. The root-logger sits at the top of the logger hierarchy, and its values are associated with each individual logging event that is not specifically associated with a child logger. The other loggers inherit the values of the root logger.

• Logging subsystem components such as handlers, loggers, and the ROOT logger can be configured and managed in several ways, using the EAP administration console or EAP CLI, or by manually editing the XML configuration files (not recommended).).

• Log levels can be changed dynamically at runtime to assist with troubleshooting in production without the need to reboot the server.