Lab 9 – Federating a Cell

At the end of the lab, you should be able to:
• Create a deployment manager profile
• Back up the deployment manager configuration
• Use the deployment manager administrative console
• Federate a node into the deployment manager cell
• Create a custom profile
• Create an unmanaged web server node
• Use the administrative console to start and stop a web server • Map an application to a web server

During this exercise, you change your stand-alone application server environment to a cell environment that contains two federated nodes and an unmanaged node for a web server. It is important as you progress through the exercise that you have a good understanding of what you are creating.

When you complete the exercise, you have a cell, named `was85hostCell01`, containing the following nodes:

- Deployment manager node, named `was85hostCellManager01`
- A federated node, named `was85hostNode01`, containing a node agent and an application server, named `server1`
- A federated node, named `was85hostNode02`, containing only a node agent
- An unmanaged node, named `ihsnode`, containing an IBM HTTP Server administrative process and a web server, named `webserver1`

Section 1: Use the Profile Management Tool to create a deployment manager profile

During this section of the exercise, you use the Profile Management Tool to create a deployment management profile. The deployment manager profile defines a cell, named `was85hostCell01`, containing a deployment manager node, named `was85hostCellManager01`. The existing application server, `server1`, continues to be a stand-alone server that is contained in the node `was85hostNode01`.

Section 2: Back up the Dmgr profile configuration

Before continuing, it is a good practice to back up the configuration for the Dmgr profile that was created.

Section 3: Federate profile1 into the cell of the deployment manager

During this section of the exercise, you federate the application server node, which profile1 defines (and is named `was85hostNode01`), into the cell named `was85hostCell01`, which the deployment manager profile defines. The federation process adds a node agent to the application server node.

Section 4: Create a custom profile and federate it into the deployment manager cell

During this section of the exercise, you are going to create a custom profile, `profile2`, that defines a node, named `was85hostNode02`. The custom profile is automatically federated into the cell `was85hostCell01`.

A custom profile is useful because it does not create any application servers on the node; it creates the configuration and the node agent only. Consequently, no server1 is created on that node. This feature is helpful for expanding clusters.

Section 5: Add the IBM HTTP Server to the cell

During this section of the exercise, you add an unmanaged node, ihsnode, to the cell was85hostCell01. You also add a web server, webserver1, to the unmanaged node. Information about the web server is communicated to the deployment manager through the IBM HTTP Server administrative process.

Create a node and add the web server to the node. When adding a node, you can create either a managed node or an unmanaged node. A managed node contains a WebSphere Application Server and a node agent. An unmanaged node does not have a node agent and is used for defining remote web servers in the topology.

Section 6: Add the web server to the configuration

In this section, the web server definition is added to the ihsnode.

Section 7: Mapping modules to servers

Each module of an application is mapped to one or more target servers. The target server can be an application server, a cluster of application servers, or a web server. Web servers that are specified as targets have the routing information for the application, which is generated in their plug-in configuration files.

This mapping usually takes place during application deployment. But since the DefaultApplication is deployed when this particular web server was added, the DefaultApplication still must be mapped to your new web server. That, in fact, is done for you during the last step of defining the web server properties when **All** is selected for the **Application mapping to the web server**. That step mapped all installed applications to the new web server.

Section 8: Working with the plug-in configuration file

The plug-in configuration file contains routing information for all applications that are mapped to the web server. The plug-in configuration file must be regenerated and propagated to the web server whenever changes that are made to the WebSphere configuration affect how requests are routed from the web server to the application server.

Section 9:Test the plug-in configuration

By default, the web server plug-in module checks for a new configuration file every 60 seconds. You can wait for the plug-in to find the changes, or you can restart the web server to pick up the changes immediately.