IBM WebSphere Portal

A Step-By-Step Guide to performing staging to production using Portal Application Archive in WebSphere Portal 8.5

David Batres
WebSphere Portal Level 2 Support
IBM Raleigh Lab

October 2015

Table of Contents

Introduction

This guide is meant to provide a comprehensive understanding of the staging to production process used in IBM WebSphere Portal 8.5.

Terminology

Managed Pages – a Portal page that has a representation in Web Content Management (WCM).
Portal Site library – the WCM library that stores managed pages.
Portal Application Archive (PAA) – a structured ZIP compression file that stores various Portal artifacts
Assembly – the top level directory in the PAA file. An assembly can contain several components.
Component – directory within the PAA file where all deployable artifacts are stored.

What is staging to production?

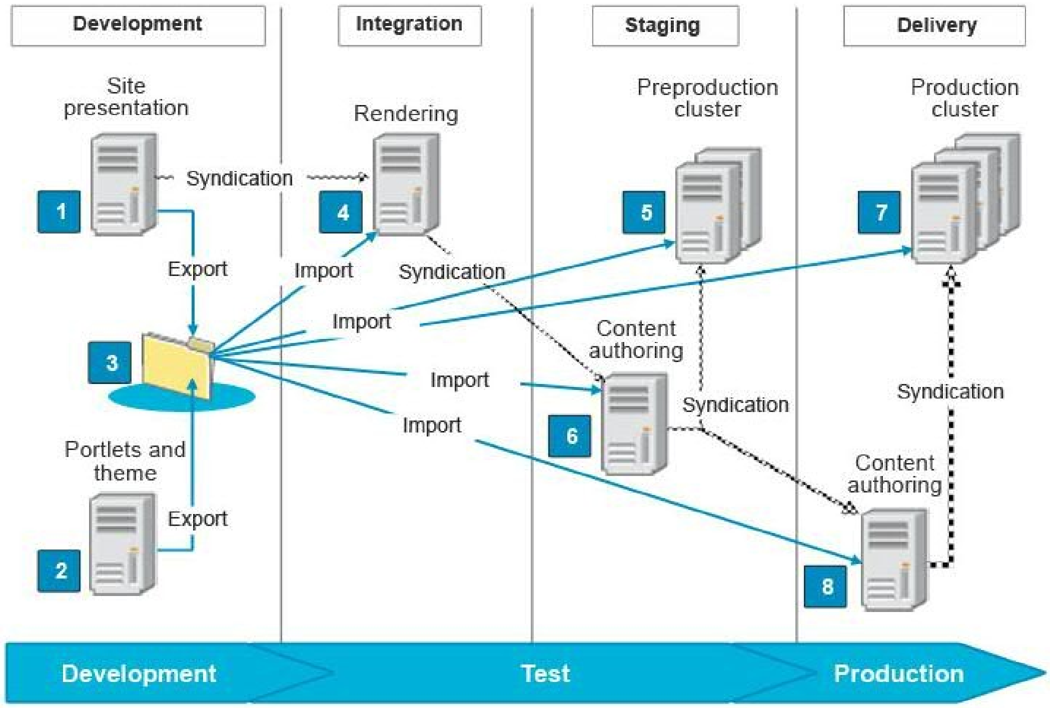
During the development of a Portal website, you initially develop, test, and refine the site on one server or a limited number of servers. You later deploy the website on live systems, referred to as the production environment. The process of moving the website from the development environment to the production environment is called staging.[1]

Staging is required at multiple times during the development cycle of a Portal website. As new artifacts are deployed, you create a new release (or version) of the site and then move it to the production system. During the development lifecycle, multiple releases of the website are created and brought into production.[1]

IBM WebSphere Portal and IBM Web Content Manager solutions can consist of many artifacts. These artifacts include portlets, themes and skins, portlet services, page layouts, wires, portlet configurations, portlet data, web content, and personalization rules. Staging helps you move those artifacts to the production environment in a controlled way.[2]

Staging can also be used to create an identical copy of a Portal website. For this guide, we will refer to the staging server as the SOURCE system, and the production server as the TARGET system. Keep in mind that source system can refer to any environment where the artifacts are developed and deployed, and target system can refer to any environment where you want to make an identical copy of the source system.

Here is a diagram showing sample applications of the staging to production process.



The following information describes the diagram annotation. Each numbered box in the diagram has a corresponding description below.

1. The web content development environment is where the following artifacts are developed:
   - Presentations
   - Authoring templates
   - Taxonomies
   - Content libraries
   - Content components such as personalization components
   - Personalization rules

2. The Portal development environment is where the following artifacts are developed:
   - Portlets
   - Extensions
   - Themes
   - Applications

3. The export directory is where you store all source artifacts, including Web Content Manager artifacts. You can either compress this directory and import it to your other staging environments or build a Portal Application Archive (PAA) file.
4. Use the rendering environment to test the rendering of the web content and portal development artifacts together. Developers should have access to this environment to test their code.
5. Use the preproduction environment to test everything on a cluster configuration that mirrors the live cluster. Only the administrators have access to this environment.

6. You can use the Content Authoring environment to test the syndication process before you move to the production environment. This environment is the first place where you can test the full content of your site. The full content includes the page structure and the web content. The authoring (6) and rendering (5) environments are also a good place to run more performance measurements. These measurements ensure that your production environment performs to your expectations. Some teams might also want to use the staging environment for training or demonstration purposes.
7. The production cluster environment is the live website cluster.
8. The authoring environment is where content developers and editors develop new content and content updates for the site.[2]

See the References section for more information.

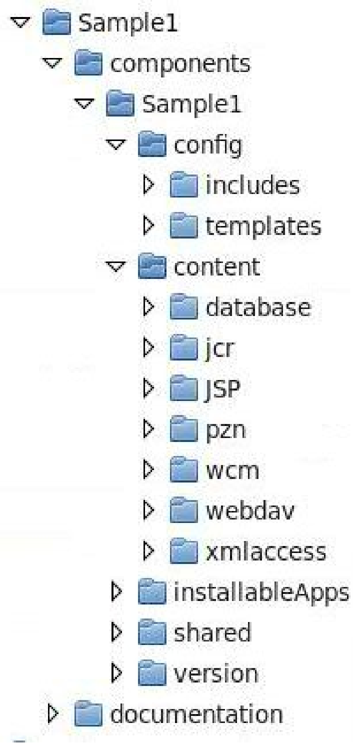
What is PAA?

PAA stands for Portal Application Archive. Each PAA file is a structured ZIP compression file.[3] It is used to package artifacts from a Portal website in a single deployable file.[4] There is a predefined structure used to package the PAA.

The components directory and its subdirectories in the PAA file archive are where all deployable artifacts must be stored. A component can include artifacts and configuration details for an entire application.[4]

The artifacts that can be deployed includes, but is not limited to:
Ant tasks
JCR-related artifacts
JSPs
Web Content libraries
WebDAV file store artifacts
XMLAccess scripts
DDL and SQL scripts to create and populate database tables
PZN-related artifacts
EAR files
WAR files
shared libraries

Here is the structure of a sample PAA provided with Portal.



```
▽ Sample1
  ▽ components
    ▽ Sample1
      ▽ config
        ▷ includes
        ▷ templates
      ▽ content
        ▷ database
        ▷ jcr
        ▷ JSP
        ▷ pzn
        ▷ wcm
        ▷ webdav
        ▷ xmlaccess
      ▷ installableApps
      ▷ shared
      ▷ version
  ▷ documentation
```

For more information about PAA, see Developing basic and advanced PAA file applications in the References section.

Environment setup

For this guide, both the source and target environments are fresh installations and not migrated from a previous version. Managed Pages is enabled on both systems before starting the staging process.

Windows/Unix Differences

This guide was written using Linux as the base operating system, however the steps/concepts listed in this guide are independent of operating system. The only significant difference is that for Windows, you must use the batch file commands instead of the UNIX shell commands listed in this guide.
For example:
UNIX: ./ConfigEngine.sh deploy-paa
Windows: ConfigEngine.bat deploy-paa

Also, it is recommended to use the Command Prompt (cmd.exe) on Windows instead of PowerShell.

File system paths

The following tags are used to reference common file system paths.

<AppServer_root> - The root path of the AppServer directory, for example:
/opt/IBM/WebSphere/AppServer

<PortalServer_root> - The root path of the PortalServer directory, for example:
/opt/IBM/WebSphere/PortalServer

<wp_profile_root> - The root path of the wp_profile directory, for example:
/opt/IBM/WebSphere/wp_profile

Troubleshooting

The following logs were used to troubleshoot errors encountered throughout the guide.

```
<wp_profile_root>/logs/WebSphere_Portal/SystemOut.log
<wp_profile_root>/logs/ffdc/<ffdc_log>.txt
<wp_profile_root>/ConfigEngine/log/ConfigTrace.log
<wp_profile_root>/ConfigEngine/properties/wkplc.properties
<wp_profile_root>/ConfigEngine/properties/wkplc_comp.properties
```

It is recommended to use the Portal Log Analyzer tool for parsing the SystemOut log and ConfigTrace log in an easy to read format. The tool can be accessed from Greenhouse here:

https://greenhouse.lotus.com/plugins/plugincatalog.nsf/assetDetails.xsp?action=editDocument&documentId=C7AB4C82F17EC6228525790B006F6D43

Preparing the source environment

The source environment used in this guide was created using the following specifications.

- IBM WebSphere Application Server Network Deployment 8.5
- IBM WebSphere Portal 8.5
- IBM Installation Manager 1.7.2
- standalone server; not a federated or clustered environment
- IBM DB2 10.5 FP3
- federated repository – fileRegistry + IBM Directory Server 6.2

Important: To avoid hitting known defects in the code, it is crucial to apply the latest WAS fix pack, and then apply the latest Portal CF. In particular, CF06 includes many fixes that improve the PAA deployment process.

When developing a Portal website on the source server, you will have pages, portlets, themes, skins, personalization rules, web content, and other artifacts. For this guide, the following changes were deployed to the SOURCE environment.

- change context root from:
    - /wps → /web
    - /portal → /public
    - /myportal → /secure
- create virtual portal with separate user realm population
- create private pages
- create custom property for resource environment providers
- deploy Portal 8.0 based EAR/WAR theme
- deploy Portal 8.5 based EAR/WAR theme
- deploy Portal 8.5 based WebDAV theme
- deploy custom application
- deploy pre-deployed application
- deploy portlet utilizing credential vault
- deploy IBM Script portlet
- configure Web Application Bridge (WAB)
- create a shared library
- configure explicit login filter
- create a Web Content library
- assign role access to Web Content library
- create custom workflow
- create personalization rules

This list contains many common customizations made to a Portal website, but is not exhaustive of all possible changes that can be made to an environment.

Preparing the target environment

You must set up your target environment before you move content from the source server to the target servers. These steps remove any content from the TARGET servers to prepare them for the following import of the release data. This allows for the content to be imported smoothly and avoid any conflicts with existing data. It is assumed that these servers are fresh installed servers. The servers must not contain any customization data.[5]

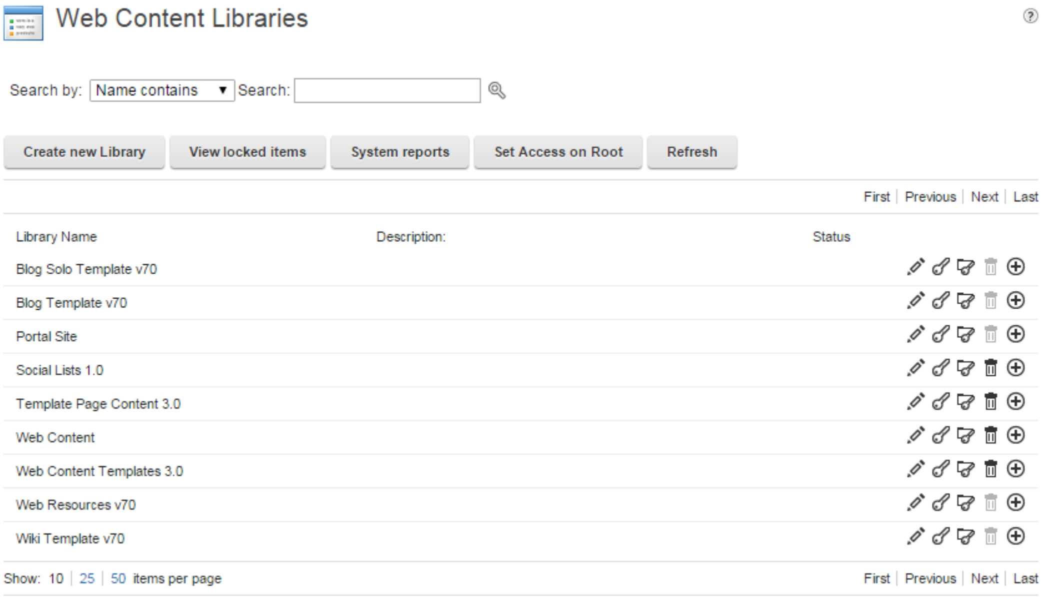
The following procedure was used to set up the target environment.

- Install the IBM WebSphere Portal server
- Apply the same Cumulative Fix (CF) that was applied to the source environment.
- Configure Portal to the desired external database
- Change the context root to match the source environment
- Cluster the target environment with additional horizontal and vertical members
- Configure the same security settings as the source environment. This includes configuring the same LDAP as the source environment.
- Create the non-default realms to be used by virtual portals

Note: The security configurations must match on the source and target environments. This is to ensure user(s) and group(s) are found when importing content. If a separate LDAP is needed, it is recommended to configure the LDAP after completing the staging to production process.

After completing the above steps, delete the existing Web Content libraries on the target environment.

1. Log into the target server
2. Click on the **Administration menu** in the toolbar
3. Click **Portal Content > Web Content Libraries**

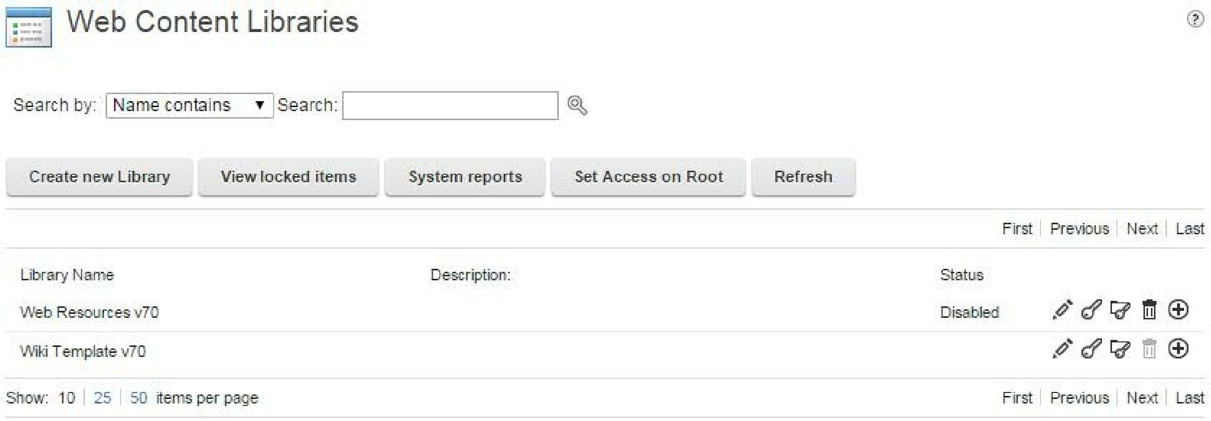


Library Name	Description:	Status
Blog Solo Template v70		
Blog Template v70		
Portal Site		
Social Lists 1.0		
Template Page Content 3.0		
Web Content		
Web Content Templates 3.0		
Web Resources v70		
Wiki Template v70		

4. Click Delete Library for each library on the system.

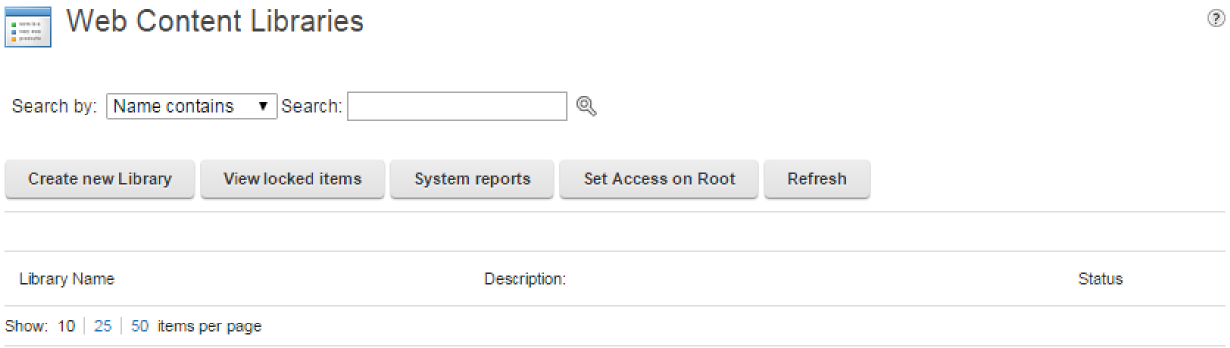
   Note: Edit any libraries that have the option **Prohibit library from being deleted.** Clear that option before you delete them.

5. After you attempt to delete a library, you may find the Status says Disabled, but the library still exists even after refreshing the page.



Library Name	Description:	Status
Web Resources v70		Disabled
Wiki Template v70		

If this happens, skip the library and come back to it after you attempted to delete all other libraries. Repeat this process until all of the libraries are deleted. The libraries may have dependencies among themselves and need to be deleted in a certain order.



Library Name	Description:	Status

If you use the web application bridge (WAB) feature, set the context root for the wp.vwat.servlet.ear application.

1. Log on to the WebSphere Integrated Solutions Console.
2. Go to **Applications > Application Types > WebSphere enterprise applications**.
3. Find and click the **wp.vwat.servlet.ear** application link.
4. Under the **Web Module Properties** heading, click **Context Root For Web Modules**.
5. Change the context root to /. This step can create name conflicts. Add a rewrite rule to avoid these conflicts.
6. Click **OK**.
7. Click **Save** to save your changes to the master configuration.
8. Stop and restart the **wp.vwat.servlet.ear** application.

Creating the initial release

In order to stage your initial release to another server, you must build the initial release Portal Application Archive (PAA) file from your source server. This can be done using the *build-initial-release-paa* ConfigEngine task.

Ensure the Portal server is started before running this command. For this guide, the following parameters were used. The command is split into multiple lines for clarity. Be sure to type out the command into the command window instead of copying directly from this guide.

```
./ConfigEngine.sh build-initial-release-paa
-DdestPAADir=/tmp
-DWasPassword=<password>
-DPortalAdminPwd=<password>
-DexportREP=true
-DexportWebDavTheme=true
-DexportWcmData=true
-DsharedAppResourcesRootDir=/sharedLib
-DappsToExtract=PAA_WARTheme,PredeployedAppEAR,custom8001ThemeEAR,scriptportlet
```

Refer to the knowledge center for additional parameters to customize the release.
http://www-01.ibm.com/support/knowledgecenter/SSHRKX_8.5.0/mp/deploy/dep_cust_paa.dita?lang=en

destPAADir this specifies the directory where the PAA will be written to. The parameter must point to an existing directory that the user has access to.

exportREP specifies to include the custom properties for *all* resource environment providers. The default is true. Set this parameter to false if you intend to manually recreate custom properties for resource environment providers.

exportWebDavTheme is set to true in order to export WebDAV resources.

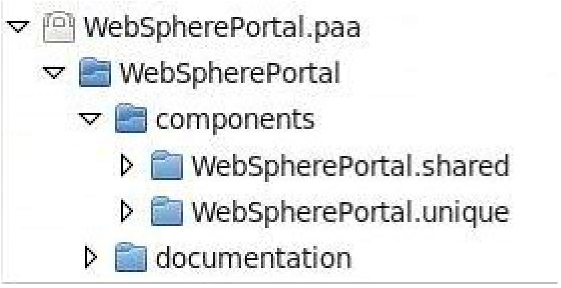
exportWcmData is set to true in order to export Web Content libraries. If your Web Content libraries are large, you should consider setting this parameter to false and using database tools to copy the JCR database as an alternative to exporting and importing the libraries.

sharedAppResourcesRootDir specifies the shared libraries used by the source environment.
Note: This will collect *all* JAR files in the specified directory. If the required JAR files span multiple directories, or only some JAR files in the specified directory are needed, it is recommended to leave this parameter empty and manually copy the JAR files into the PAA. See the below screenshots for a breakdown of where these JAR files are stored within the PAA.

appsToExtract specifies the EAR files that are deployed to the source environment. These EAR files are typically used for theme resources and pre-deployed applications.

The result of running *build-initial-release-paa* will generate WebSpherePortal.paa used for the base portal, and a PAA file for every virtual portal in the source environment. The name of the PAA for the virtual portal is based on the objectID of that virtual portal. For example, we will use Z18_G1ACH880LO5O80AQ7JG5J63000 as an example of the virtual portal objectID throughout this guide, but your virtual portal objectID may differ.

The WebSpherePortal.paa file contains artifacts in specific directories that are used to deploy on the target environment. Within this PAA, the assembly name is WebSpherePortal. And within the WebSpherePortal assembly are two components:
WebSpherePortal.shared
WebSpherePortal.unique

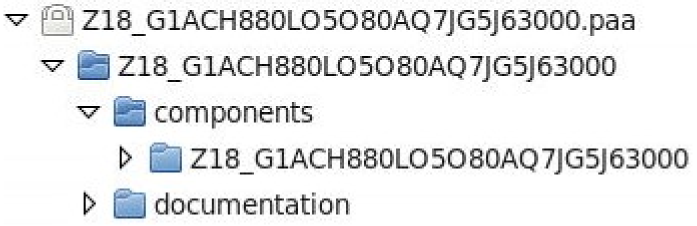


```
WebSpherePortal.paa
  WebSpherePortal
    components
      WebSpherePortal.shared
      WebSpherePortal.unique
    documentation
```

The WebSpherePortal.shared component contains artifacts that are shared between the base and virtual portals. This includes applications, Personalization (PZN) rules, credential vault slots, WebDAV resources, resource environment providers custom properties, and shared libraries.

The WebSpherePortal.unique component contains artifacts that are unique to the base portal. This includes non-managed pages and Web Content libraries.

The WebSpherePortal.shared component is needed for both base and virtual portals. However, the WebSpherePortal.unique is only needed for the base portal. If your intention was to only stage virtual portals from the source environment, you can specify *-DexportUniquePortalPAA=false* on the command line when building the initial release to exclude the base portal unique component.

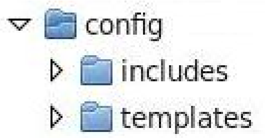


```
Z18_G1ACH880LO5O80AQ7JG5J63000.paa
  Z18_G1ACH880LO5O80AQ7JG5J63000
    components
      Z18_G1ACH880LO5O80AQ7JG5J63000
    documentation
```

Within the virtual portal paa, Z18_G1ACH880LO5O80AQ7JG5J63000.paa, the assembly and component names are based on the objectID of the virtual portal. This PAA only contains one component that stores the unique artifacts for that virtual portal.

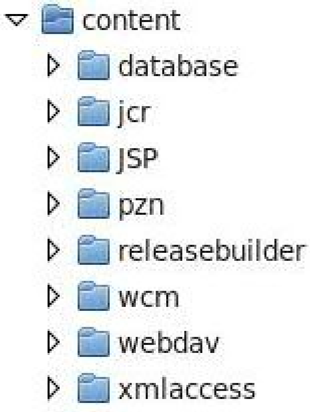
There are several directories in each component that are used to store the artifacts. A few directories will be highlighted in this guide. See the Components directory[4] link in the References section for a complete listing of the purpose for each directory if further customization is needed.

Within each component...



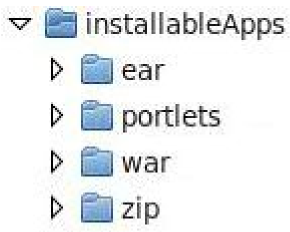
```
▽ config
    ▷ includes
    ▷ templates
```

- The config/includes directory stores ConfigEngine tasks used to deploy the content. Custom ConfigEngine task extensions can be placed here.
- The config/templates directory stores extra script files used by ConfigEngine tasks. For example, XML files used to deploy a WAR file can be placed here. These scripts are not called automatically and must be explicitly called from within a custom ConfigEngine task.

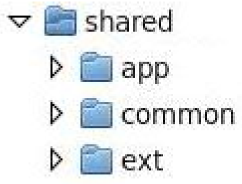


```
▽ content
    ▷ database
    ▷ jcr
    ▷ JSP
    ▷ pzn
    ▷ releasebuilder
    ▷ wcm
    ▷ webdav
    ▷ xmlaccess
```

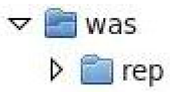
- The content/pzn directory stores PZN rules as NODES files, along with PZN JAR files.
- The content/releasebuilder directory stores XML files to be used with releasebuilder when creating a differential PAA.
- The content/wcm directory stores Web Content libraries.
- The content/webdav directory and its subdirectories store WebDAV related artifacts. There are four possible subdirectories; each one named to reflect the type of function that is provided in the files supplied: iwidgets, layout-templates, skins, themes. The *deploy-paa* ConfigEngine task (used later in this guide) will deploy the artifacts in each directory to the corresponding WebDAV entry point. For example, themes.zip within the content/webdav/themes folder will be deployed to fs-type1/themes WebDAV entry point.
- The content/xmlaccess directory stores installation scripts that are called automatically with XMLAcccess.



- The installableApps/ear directory stores EAR files that were specified with -DappsToExtract.
- The installableApps/war directory stores WAR files that are not automatically installed. A developer would deploy any WAR files stored here with a custom Ant task to install the artifacts. The installer generally ignores this directory apart from copying the files to <wp_profile_root>/installableApps and then starting any custom code that is provided to handle these artifacts.



- The shared/app directory stores libraries used for custom portlets, login filters, etc. The contents of this directory will be registered in this JAR, which is read during Portal startup: <wp_profile_root>/PortalServer/solutionInstaller/shared/app/shared.app.jar
- If you did not specify -DsharedAppResourcesRootDir on the command line, you can manually copy your JAR files in the shared/app directory within the WebSpherePortal.shared component of WebSpherePortal.paa.



- The was/rep directory stores custom properties for resource environment providers.
  Note: if you included custom properties for resource environment providers, review the properties files in this directory to validate the values for use in the target environment. Consider updating any properties related to hostnames, cell names, or other configurations specific to the source environment. For example, host.name or redirect.logout.url within WP ConfigService.properties may need to be changed for the target environment.

Deploying the initial release

After you create and prepare your initial source and target environments, you will deploy the contents of the source server to the target environment. If you have a cluster, deploy on the primary node of the cluster before deploying to the secondary nodes. Review the Known Problems section if you encounter any issues.

1. Take a backup of the target environment. This includes the file system and database at a minimum.
2. Copy all of the PAA files to a temporary location on the primary node of the target environment.
3. Ensure the Deployment Manager is started.
4. Ensure that automatic synchronization is disabled on all secondary nodes and that all node agents are stopped on the target environment. When automatic synchronization is enabled, the node agent on each node automatically contacts the Deployment Manager at startup and at every synchronization interval to attempt to synchronize the node's configuration repository with the master repository managed by the Deployment Manager.

   Important: Ensure automatic synchronization is enabled *only* on the primary node and disabled on secondary nodes.

   a) In the administrative console for the Deployment Manager on the target environment, select **System administration > Node agents** in the navigation tree.
   b) Click the **nodeagent** link for a secondary node.
   c) Click **File Synchronization Service**.
   d) Uncheck the **Automatic Synchronization** check box and uncheck the **Enable service at server startup** check box on the File Synchronization Service page to disable the automatic synchronization feature and synchronization service at server startup and then click **OK**.
   e) Click **Save** to save the configuration changes to the master repository.
   f) Repeat these steps for all other secondary nodes in the target environment.
   g) Select **System administration > Nodes** in the navigation tree
   h) Select all nodes that are not synchronized, and click on **Full Resynchronize**. The nodeagents will need to be started for this to work.
   i) Select **System administration > Node agents** in the navigation tree
   j) For the primary node and all additional nodes, select the nodeagents and click **Stop**.

5. On the primary node, ensure the nodeagent and Portal servers are stopped.
6. Open a command window and navigate to <wp_profile_root>/ConfigEngine
7. Run the *install-paa* ConfigEngine task for the base portal PAA. For example,

   ```
   ./ConfigEngine.sh install-paa
   -DPAALocation=/tmp/WebSpherePortal.paa
   -DWasPassword=<password>
   -DPortalAdminPwd=<password>
   -Dwp.si.offlineMode=true
   ```

   The *install-paa* task lays down the PAA binaries and registers the components with ConfigEngine. On a lab machine, this took less than one minute to complete.

8. Run the *deploy-paa* ConfigEngine task for the base portal PAA. For example,

   ```
   ./ConfigEngine.sh deploy-paa
   -DappName=WebSpherePortal
   -DforceDeploy=true
   -DWasPassword=<password>
   -DPortalAdminPwd=<password>
   ```

   Depending on the amount of artifacts being deployed and performance of the system, this may take a long time to complete. On a lab machine, this took 1hr 30 min to complete.

9. Run the *activate-portlets* ConfigEngine task to ensure the applications are active.

   ```
   ./ConfigEngine.sh activate-portlets
   -DWasPassword=<password>
   -DPortalAdminPwd=<password>
   ```

   On a lab machine, this took less than one minute to complete.

10. On the primary node, run the *install-paa* ConfigEngine task for the virtual portal PAA. For example,

    ```
    ./ConfigEngine.sh install-paa
    -DPAALocation=/tmp/Z18_G1ACH880LO5O80AQ7JG5J63000.paa
    -DWasPassword=<password>
    -DPortalAdminPwd=<password>
    -Dwp.si.offlineMode=true
    ```

    On a lab machine, this took less than one minute to complete.

11. Run the *list-all-virtual-portals* ConfigEngine task on the SOURCE environment to determine the hostname or context associated with the virtual portal PAA. For example,

    ```
    ./ConfigEngine.sh list-all-virtual-portals
    -DWasPassword=<password>
    -DPortalAdminPwd=<password>
    ```

    Note: this ConfigEngine task needs to be run on the SOURCE environment to verify the correct virtual portal parameters. You can determine the objectID and virtual portal parameters by inspecting the output on the command window, or reviewing wp_profile/ConfigEngine/log/ConfigTrace.log.

Here is a sample output of running *list-all-virtual-portals*:

```
[wsadmin] ______________________________________________________
[wsadmin] VirtualPortal:
[wsadmin] Title:       paawp
[wsadmin] Description: Virtual Portal created just for the whitepaper project
[wsadmin] Realm:       paaVPRealm
[wsadmin] Object ID:   Z18_G1ACH880LO5O80AQ7JG5J63000
[wsadmin] Hostname:    <cannot be retrieved>
[wsadmin] Context:     VirtualPortal1
[wsadmin] Short ID:    32262
[wsadmin] ______________________________________________________
```

12. Run the *deploy-paa* ConfigEngine task on the target environment for the virtual portal PAA. The syntax will be different between a hostname-based virtual portal and a context-based virtual portal.

    a) If you are deploying a hostname-based virtual portal, use this syntax:

    ```
    ./ConfigEngine.sh deploy-paa
    -DappName=<vp_object_id>
    -DforceDeploy=true
    -DWasPassword=<password>
    -DPortalAdminPwd=<password>
    -DVirtualPortalHostName=myvirtualportal.mycompany.com
    ```

    b) If you are deploying a context-based virtual portal, use this syntax:

    ```
    ./ConfigEngine.sh deploy-paa
    -DappName=<vp_object_id>
    -DforceDeploy=true
    -DWasPassword=<password>
    -DPortalAdminPwd=<password>
    -DVirtualPortalContext=VirtualPortal1
    ```

    On a lab machine, this took 5 minutes to complete.

13. Repeat steps 10-12 on the primary node for each virtual portal PAA.
14. On the secondary node in the cluster, ensure the nodeagent and Portal servers are stopped.
15. Copy the WebSpherePortal.paa to a temporary location on the secondary node. You do NOT need to copy the virtual portal PAA files to the secondary node.

16. Open a command window and navigate to: <wp_profile_root>/ConfigEngine.
17. Run the *install-paa* ConfigEngine task for the base portal PAA. For example,

    ```
    ./ConfigEngine.sh install-paa
    -DPAALocation=/tmp/WebSpherePortal.paa
    -DWasPassword=<password>
    -DPortalAdminPwd=<password>
    -Dwp.si.offlineMode=true
    ```

    On a lab machine, this took less than one minute to complete.

18. Run the *deploy-paa* ConfigEngine task for the base portal PAA. For example,

    ```
    ./ConfigEngine.sh deploy-paa
    -DappName=WebSpherePortal
    -DforceDeploy=true
    -DWasPassword=<password>
    -DPortalAdminPwd=<password>
    ```

    On a lab machine, this took 5 minutes to complete.

19. Repeat steps 14-18 on every secondary node in the cluster.
20. Re-enable automatic synchronization on all nodes in the cluster.

    a) In the administrative console for the Deployment Manager, select **System administration > Node agents** in the navigation tree.
    b) Click the **nodeagent** link for the secondary node.
    c) Click **File Synchronization Service**.
    d) Check the **Automatic Synchronization** check box to enable the automatic synchronization feature and check the **Enable service at server startup** check box to enable the synchronization service at server startup on the File Synchronization Service page and then click **OK**.
    e) Click **Save** to save the configuration changes to the master repository.
    f) Repeat these steps for all remaining nodes.
    g) Select **System administration > Nodes** in the navigation tree
    h) Select all nodes that are not synchronized, and click on **Full Resynchronize**. The nodeagents will need to be started for this to work.
    i) Select **System administration > Node agents** in the navigation tree
    j) Select all node agents where automatic synchronization has been re-enabled and click **Restart**.

21. Review the section “More components to stage” for any items that are not covered by the PAA. Export or manually re-create any items that are not handled by the initial import.

    http://www-01.ibm.com/support/knowledgecenter/SSHRKX_8.5.0/mp/deploy/dep_stage_check.dita?lang=en

22. If you use Web Services for Remote Portlets (WSRP), review the configuration of the WSRP web services.
    a) If you use your portal as a WSRP producer, manually re-create the web service security configuration of the WSRP service providers. For more information, see the WSRP services section.
    b) If you use your portal as a WSRP Consumer, verify the web service security configuration and manually re-create the web service security configuration if needed.
        i. If you configured WSRP Producer ports for web service security, for example, by using the Web Services admin portlet or XMLAccess, review the web service security configuration.
        ii. If you configured WSRP web service clients, for example, by configuring WSRP service clients through policy sets in the administrative console or by adding service references, you must manually re-create the configuration of the WSRP web service clients. For more information, see the WSRP services section.
        iii. If you configured web service security by using an LTPA version 1 token, you can enable those tokens. For more information, see the WSRP services section.
        iv. If you configured HTTP-cookie-based single sign-on, for example, by configuring corresponding client cookie forwarding rules, you must review and re-create the resource environment properties for the cookie forwarding rules. For more information, see the WSRP services section.

23. *Optional:* Export search collections from the source server and import them into the target server. For more information, see Exporting and importing search collections. If you are using alternative context root URLs, go to **Manage Search** portlet. To open the **Manage Search** portlet, click the **Administration menu** icon. Then, click **Search Administration > Manage Search**. Update the search collection URL links on the source server to point to the alternative root. For example,

    http://yourdomain.com:port/your_root/seedlist/myserver?SeedlistId=&Source=com.ibm.workplace.wcm.plugins.seedlist.retriever.WCMRetrieverFactory&Action=GetDocuments

24. If you installed any additional PAAs on the source server, they are not included in the initial release PAA. Each additional PAA that was installed and deployed on the source server must be manually installed and deployed on the target server after the initial release PAA is installed.
25. Review and recreate any HTTP proxy configuration profiles under **Administration > Portlet Management > Virtual Web Application Manager**. Refer to the Portal knowledge center here:

    http://www-01.ibm.com/support/knowledgecenter/SSHRKX_8.5.0/help/panel_help/h_wab_first.dita?lang=en

26. Restart the cluster.
27. **Important**: Create a copy of this file and store it in a safe location. Rename the copied file to REV1_base.xml

    <wp_profile_root>/paa/WebSpherePortal/components/WebSpherePortal.shared/content/releasebuilder/staging-version1.xml

Updates with syndication

You can use syndication to update content that was originally created by deploying the portal artifacts using a Portal Application Archive (PAA) file.

Important: Successful syndication requires that the object IDs for portlets, themes, and other artifacts are the same on both the syndicator and subscriber. Because the syndication process itself does not manage these artifacts, you must synchronize the two servers by an initial staging process before you syndicate. In addition, each time that you deploy an artifact that is not managed by syndication to the source environment, you must stage the artifact with the appropriate staging tool.[6] For example, portlets can be staged using the XMLAccess tool and theme artifacts can be staged using a WebDAV client.

Limitations:

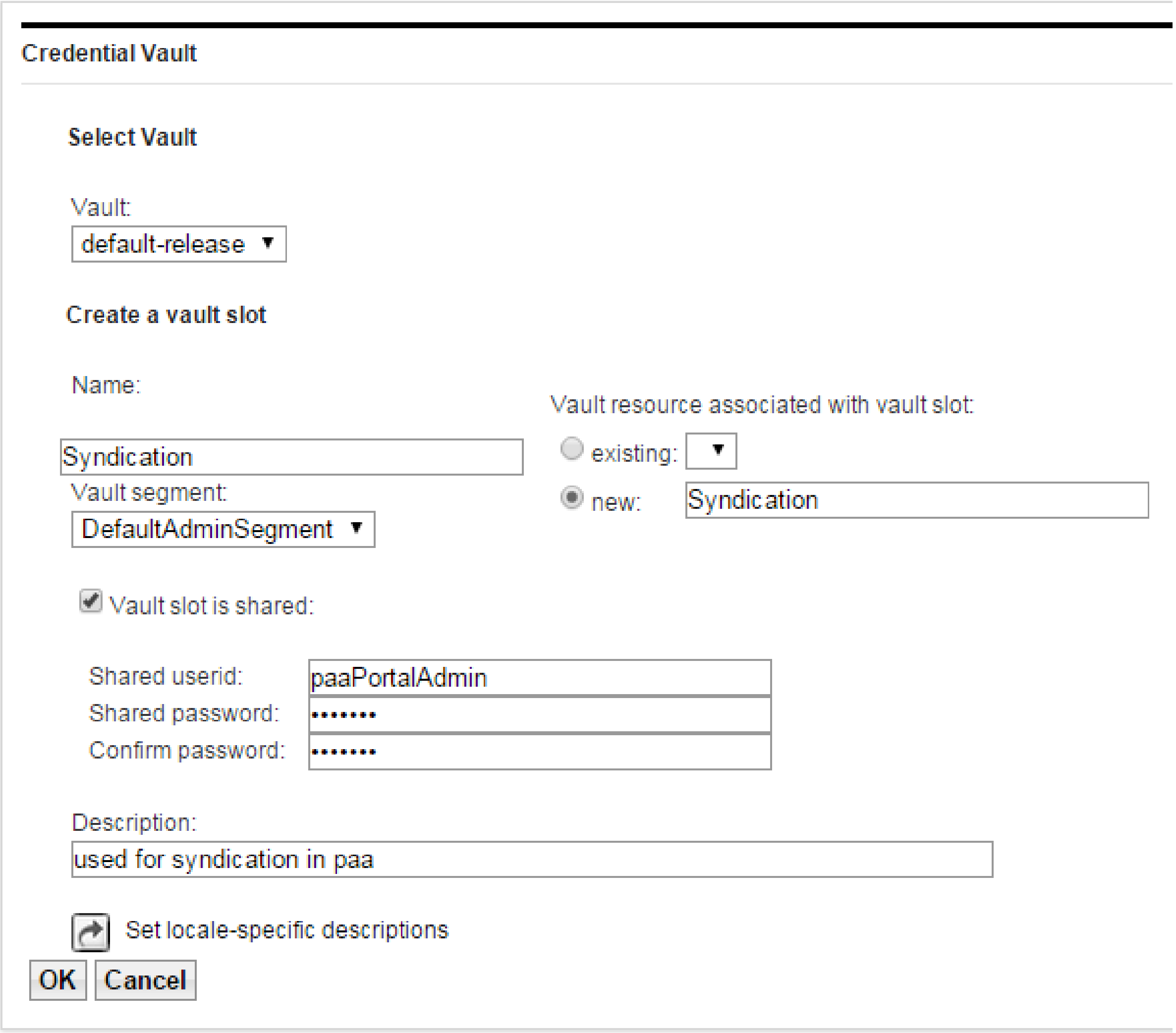
- Syndication can be set up only between systems that use the same user repository.
- Syndication for managed pages between multiple servers requires that you run an initial staging to all the servers.
- The syndication process runs a prerequisite check on the subscriber to ensure that any required themes, skins, portlets, or iWidgets on a page are present on the target system. If any required objects are missing on the subscriber, syndication is not run for the page. This behavior can result in missing pages and syndication errors for affected pages.
- iWidgets and portlets can store data in the WebDAV file store. The syndication process does not verify or transfer WebDAV data. If a portlet on a managed page requires data from the WebDAV file store, you must manually copy the required objects to the target system.[6]

If Managed Pages is enabled and you are using syndication as part of the staging process, the following considerations apply:

- In addition to web content, syndication includes artifacts such as pages and wires.
- When you populate the target server for the first time, you must populate the pages initially through a Portal Application Archive (PAA) file.
- When you update the target server after the initial staging, do not continue to export pages and wires, but instead use syndication to transfer the updates. If you export pages and wires after the initial staging, you might inadvertently overwrite changes that are already published through syndication.[6]

Use the following procedure to set up the syndication relationship. The source environment will be the syndicator, and the target environment will be the subscriber. The source environment syndicates updates to the target environment.

1. Ensure both the syndicator and subscriber are running and that they can access each other over a network.
2. On subscriber (target), log in to WebSphere Portal as an administrator.
3. Navigate to: **Administration > Access > Credential Vault**
4. Select **Add a vault slot**
5. Enter values for the slot name and vault resource.
6. Select the box to mark the slot as shared and enter the Portal admin ID and password for the subscriber, and add an optional description.
7. Click **OK**.



Credential Vault

Select Vault

Vault: default-release ▼

Create a vault slot

Name: Syndication

Vault segment: DefaultAdminSegment ▼

Vault resource associated with vault slot:

○ existing: ▼

● new: Syndication

☑ Vault slot is shared:

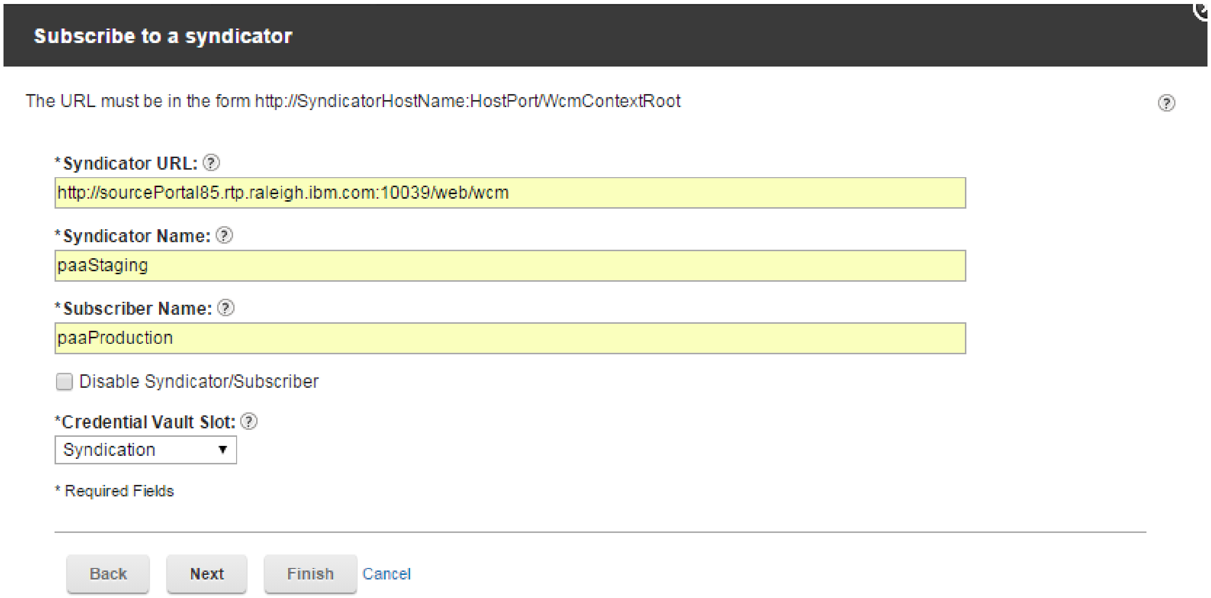
Shared userid: paaPortalAdmin

Shared password: •••••••

Confirm password: •••••••

Description: used for syndication in paa

8. Navigate to: **Administration > Portal Content > Subscribers**
9. Click **Subscribe Now**
10. In the **Syndicator URL** field, enter the syndicator URL in the form of http://HostName:HostPort/WcmContextRoot. For example, http://sourcePortal85.rtp.raleigh.ibm.com:10039/web/wcm. (Remember the Portal context root was changed from /wps to /web).
11. In the **Syndicator Name** field, enter a name for the syndicator item. This name is used for the syndicator item that is created on the syndicator server. Enter a name that helps identify the syndication relationship you are creating. This name must be unique and cannot be the same as an existing syndicator name. For example, paaStaging.
12. In the **Subscriber Name** field, enter a name for the subscriber item. This name is used for the subscriber item that is created on the subscriber server. Enter a name that helps identify the syndication relationship you are creating. This name must be unique and cannot be the same as any existing subscriber name. For example, paaProduction.
13. Select the **Credential Vault Slot** you created previously.
14. Click **Next**.



Subscribe to a syndicator

The URL must be in the form http://SyndicatorHostName:HostPort/WcmContextRoot

*Syndicator URL: http://sourcePortal85.rtp.raleigh.ibm.com:10039/web/wcm

*Syndicator Name: paaStaging

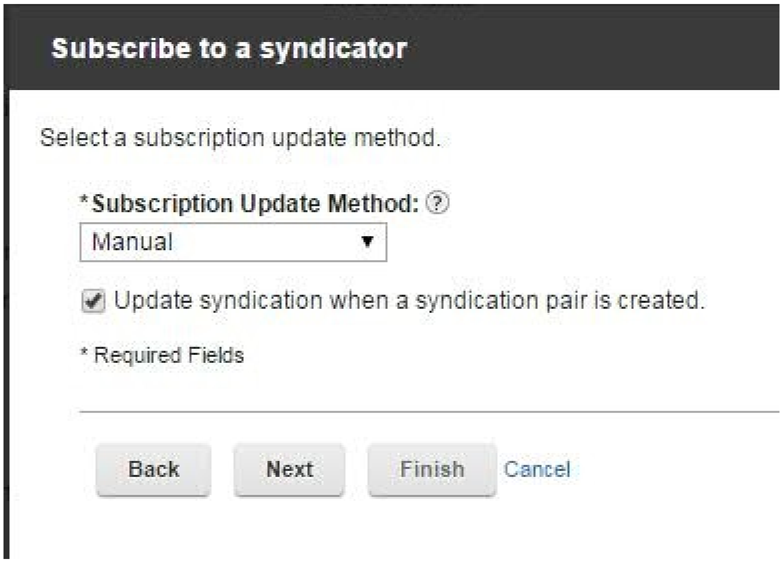
*Subscriber Name: paaProduction

□ Disable Syndicator/Subscriber

*Credential Vault Slot: Syndication

* Required Fields

15. Select Manual for the **Subscription Update Method**.



Subscribe to a syndicator

Select a subscription update method.

*Subscription Update Method: Manual

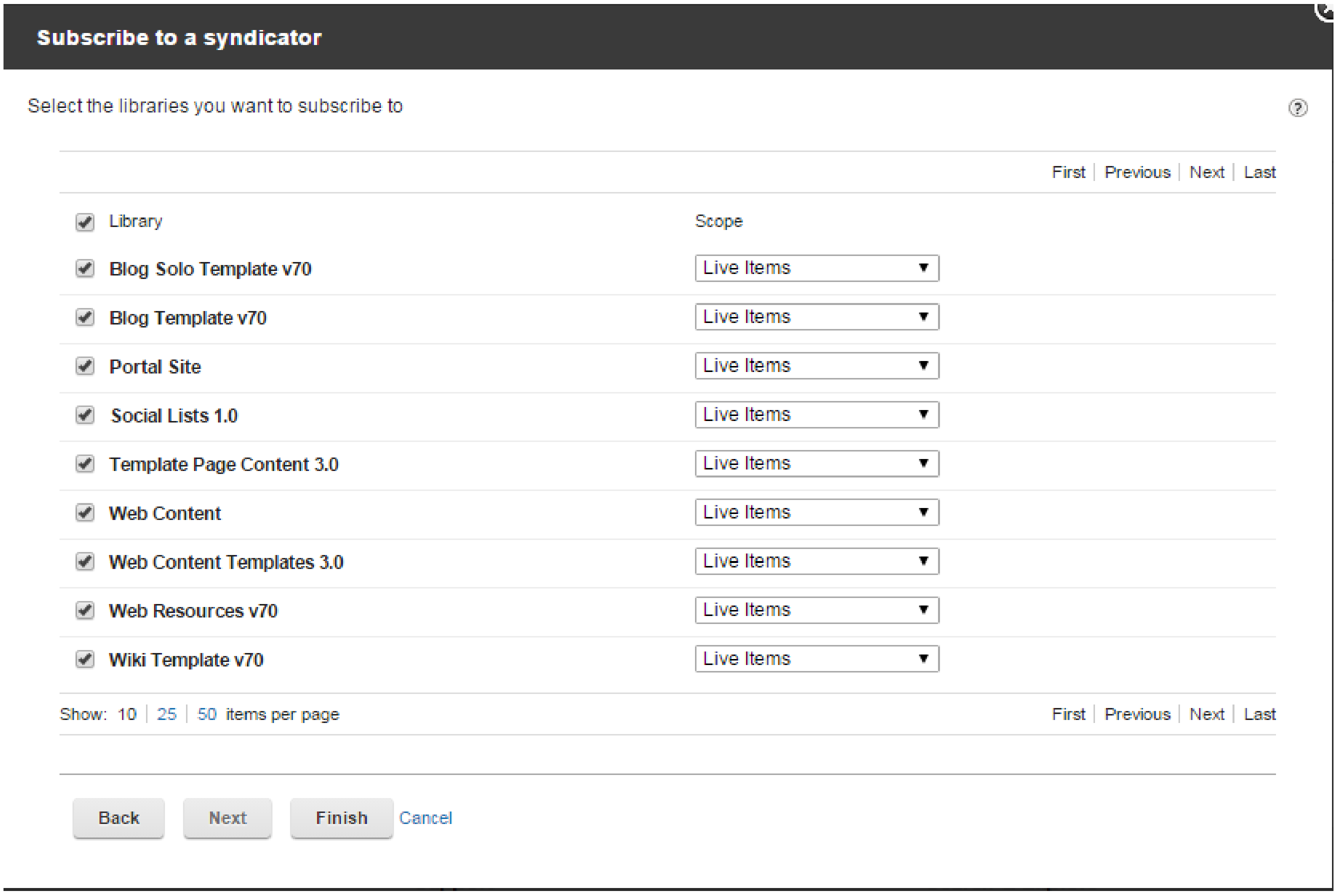
☑ Update syndication when a syndication pair is created.

* Required Fields

16. Select which libraries to subscribe to.

    Note: Depending on your business requirements, you may decide to create a syndication relationship only for the Portal Site library and set the update method to Manual, and create a separate syndication relationship for all other libraries with Automatic for the update method. For this whitepaper, we will create one syndication pair for all libraries and use Manual update method.

17. Click **Finish**.



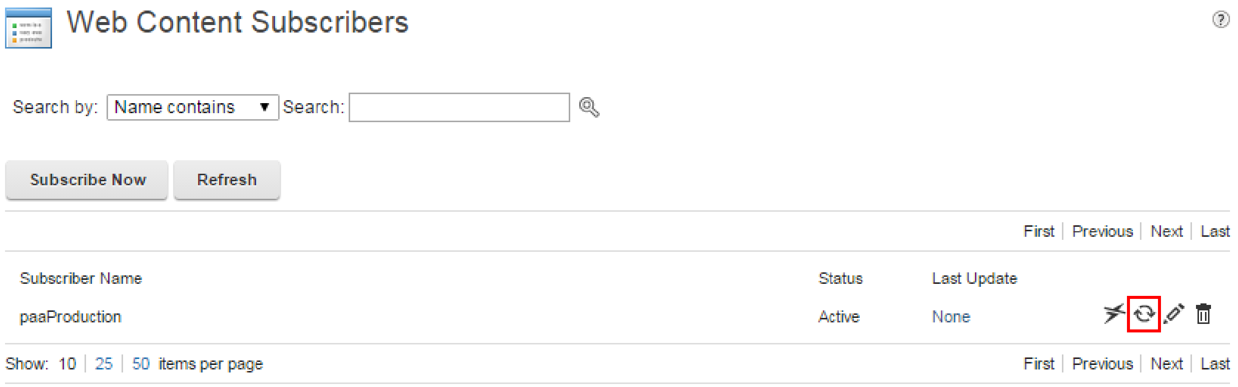
Subscribe to a syndicator

Select the libraries you want to subscribe to

☑ Library	Scope
☑ Blog Solo Template v70	Live Items ▼
☑ Blog Template v70	Live Items ▼
☑ Portal Site	Live Items ▼
☑ Social Lists 1.0	Live Items ▼
☑ Template Page Content 3.0	Live Items ▼
☑ Web Content	Live Items ▼
☑ Web Content Templates 3.0	Live Items ▼
☑ Web Resources v70	Live Items ▼
☑ Wiki Template v70	Live Items ▼

First | Previous | Next | Last
Show: 10 | 25 | 50 items per page
First | Previous | Next | Last
Back
Next
Finish
Cancel

18. To begin syndication, click the **Update Subscriber** button to the right of the subscriber name.



Web Content Subscribers

Search by: Name contains ▼ Search:

Subscriber Name	Status	Last Update
paaProduction	Active	None

Subscribe Now
Refresh
First | Previous | Next | Last
Show: 10 | 25 | 50 items per page
First | Previous | Next | Last

Updates with ReleaseBuilder

After performing the initial staging and updates with syndication, you will need to stage artifacts that are not captured through syndication, such as portlets and themes. You can use a tool called ReleaseBuilder to compare a previous release of the source environment with the current release. The differences are captured in an output file from the ReleaseBuilder tool and will be imported to the target environment. The differences can include new artifacts, updates or modifications, and deletions. In this manner, the source and target environments will remain in sync.

The ReleaseBuilder tool itself does NOT make any changes to the source or target environments. You may NOT use ReleaseBuilder to compare two releases from different environments. You MUST use ReleaseBuilder to compare two releases from the SAME environment.

In the following steps, you will need to use the REV1_base.xml file that was copied as the last step of the initial staging with PAA. As a reminder, the REV1_base.xml file was an exact copy of this file from the TARGET environment.

```
<wp_profile_root>/paa/WebSpherePortal/components/WebSpherePortal.shared/
content/releasebuilder/staging-version1.xml
```

This file will contain the initial release. A new export of the current release will be created and named REV2_base.xml. The ReleaseBuilder tool will compare REV2 with REV1 and create an XML file that captures the differences, DIFF.xml.

1. Copy the REV1_base.xml file to a temporary location on the source environment.
2. From the <wp_profile_root>/PortalServer/bin directory, create a new XMLAccess export from the SOURCE environment of the base portal. Remember: the context root was changed from /wps to /web and will be reflected in the XMLAccess comand. Use this example for reference:

   ```
   ./xmlaccess.sh
   -user <PortalAdminId>
   -password <PortalAdminPwd>
   -url http://SOURCE.portal.com:10039/web/config
   -in <PortalServer_root>/doc/xml-samples/ExportManagedPagesRelease.xml
   -out /tmp/REV2_base.xml
   ```

3. From the same directory, run the ReleaseBuilder tool for the base portal as follows:

   ```
   ./releaseBuilder.sh
   -inOld /tmp/REV1_base.xml
   -inNew /tmp/REV2_base.xml
   -out /tmp/DIFF1_base.xml
   ```

4. Store a copy of REV2_base.xml in a safe location. It will be used to deploy future updates to the TARGET environment.
5. Backup the database and file system of the entire TARGET clustered environment.
6. Copy the DIFF1_base.xml file from SOURCE to a temporary location on the TARGET environment.

7. Copy the portlets from SOURCE to the primary node of the TARGET environment. The directory name which contains the portlets is:
   <wp_profile_root>/PortalServer/deployed/archive

   This should be populated on both SOURCE and TARGET. Copy the entire directory over from SOURCE, overwriting existing contents on TARGET.

8. Using the XMLAccess command, import the DIFF1_base.xml file for the TARGET environment. For example,

   ```
   ./xmlaccess.sh
   -user <PortalAdminId>
   -password <PortalAdminPwd>
   -url http://TARGET.portal.com:10039/web/config
   -in /tmp/DIFF1_base.xml
   -out /tmp/baseimportresults.xml
   ```

9. Run the *activate-portlets* ConfigEngine task on the TARGET environment. For example,

   ```
   ./ConfigEngine.sh activate-portlets
   -DWasPassword=<password>
   -DPortalAdminPwd=<password>
   ```

10. Validate and test the results on the TARGET environment.

The above procedure can be repeated again to stage another update in the future. You will instead use REV2 as the baseline and create a new export from SOURCE named REV3. Use ReleaseBuilder to compare REV3 with REV2 and create a new differential file to import, DIFF2.xml.

Review the staging to production check list for any artifacts that were not transferred through syndication or ReleaseBuilder. For example, exporting artifacts from the WebDAV file store or using pznload script to stage personalization rules.

http://www-01.ibm.com/support/knowledgecenter/SSHRKX_8.5.0/mp/deploy/dep_stage_check.dita

Known Problems

This section will describe problems encountered throughout the staging to production process with the corresponding solution. For more problem / solution pairs identified in the process, visit the dWAnswers forum and search for keywords: digexp, portal, staging, staging-to-production

https://developer.ibm.com/answers/search.html?f=&type=question&redirect=search%2Fsearch&sort=relevance&q=%2B%5Bdigexp%5D+%2B%5Bportal%5D+%2B%5Bstaging%5D+%2B%5Bstaging-to-production%5D

Problem: Running *build-initial-release-paa* may fail on XMLAccess with 'Connection refused' message.

In ConfigTrace.log:

```
Target started: build-initial-portal-release-paa-shared-do

[xmlaccess] EJPXB0006I: Connecting to URL
http://sourcePortal85.rtp.raleigh.ibm.com:10039/web/config
[xmlaccess] EJPXB0004I: Writing output file
/opt/IBM/WebSphere/wp_profile/ConfigEngine/config/work/paa/WebSpherePortal/components/WebSpherePortal.shared/content/xmlaccess/install/staging-version1.xml
[xmlaccess] EJPXB0009E: Could not connect to portal.
[xmlaccess] java.net.ConnectException: Connection refused
[xmlaccess] EJPXB0016E: An error occurred on the client: Connection refused

BUILD FAILED
/opt/IBM/WebSphere/PortalServer/solutionInstaller/wp.si/config/includes/bld_paa_cfg.xml:425:
EJPXB0016E: An error occurred on the client: Connection refused

Total time: 6 seconds
```

Solution:

The Portal server needs to be started for XMLAccess to work. The ConfigEngine script used to build the PAA does not attempt to start Portal. Therefore, the Portal server will need to be started manually, then rerun the *build-initial-release-paa* task.

Also double check that the following properties in wkplc.properties and wkplc_comp.properties are correctly pointing to the Portal server. Use the fully qualified hostname if localhost does not work.

WpsHostName
WpsHostPort
XmlAccessHost
XmlAccessPort

Problem: Running *build-initial-release-paa* may fail during execution of releasebuilder.

In the ConfigTrace.log:

```
   [logmsg] [02/09/15 10:07:39.503 CST] EJPCA3100I: Configuration task "Exporting shared release artifacts" in progress
[xmlaccess] EJPXB0006I: Connecting to URL http://sourcePortal85.rtp.raleigh.ibm.com:10039/web/config
[xmlaccess] EJPXB0004I: Writing output file /opt/IBM/WebSphere/wp_profile/ConfigEngine/config/work/paa/WebSpherePortal/components/WebSpherePortal.shared/content/xmlaccess/install/staging-version1.xml
[xmlaccess] EJPXB0002I: Reading input file /opt/IBM/WebSphere/PortalServer/doc/xml-samples/ExportManagedPagesRelease.xml
[xmlaccess] EJPXB0020I: The request was processed successfully on the server.
    [mkdir] Created dir: /opt/IBM/WebSphere/wp_profile/ConfigEngine/config/work/paa/WebSpherePortal/components/WebSpherePortal.shared/content/releasebuilder
     [copy] Copying 1 file to /opt/IBM/WebSphere/wp_profile/ConfigEngine/config/work/paa/WebSpherePortal/components/WebSpherePortal.shared/content/releasebuilder
     [exec] Licensed Materials - Property of IBM, 5724-E76, 5724-E77, 5724-I29 and 5655-Y16, (C) Copyright IBM Corp. 2001, 2014 - All Rights reserved. US Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
     [exec] com.ibm.utilities.collections.stl.Functional$Serializer
     [exec] com.ibm.utilities.collections.stl.Functional$Serializer
```

Running <wp_profile_root>/PortalServer/bin/releasebuilder.sh manually may result in this output instead of showing releasebuilder usage:

```
[root@targetPortal85 bin]# ./releasebuilder.sh
Licensed Materials - Property of IBM, 5724-E76, 5724-E77, 5724-I29 and 5655-Y16, (C) Copyright IBM Corp. 2001, 2014 - All Rights reserved. US Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
com.ibm.utilities.collections.stl.Functional$Serializer
com.ibm.utilities.collections.stl.Functional$Serializer
```

Solution:
Remove the following two JAR files from <AppServer_root>/lib . They are only needed for the AppServer binaries of a remote Deployment Manager profile.

<AppServer_root>/lib/wkplc.comp.registry.jar
<AppServer_root>/lib/wp.base.jar

Problem: *deploy-paa* fails with WASX7023E error – SOAP connector not available

In the ConfigTrace.log:

```
===
  [wsadmin] WASX7023E: Error creating "SOAP" connection to host
"targetPortal85.rtp.raleigh.ibm.com"; exception information:
com.ibm.websphere.management.exception.ConnectorNotAvailableException: [SOAPException:
faultCode=SOAP-ENV:Client; msg=Unable to find a valid IP for host
targetPortal85.rtp.raleigh.ibm.com]
  [wsadmin] WASX7213I: This scripting client is not connected to a server process; please refer to the
log file /opt/IBM/WebSphere/wp_profile/logs/wsadmin.traceout for additional information.
  [wsadmin] WASX8011W: AdminTask object is not available.
  [wsadmin] WASX7031I: For help, enter: "print Help.help()"
     [echo] WRONG_WAS_SERVER_CREDENTIAL
     [echo] 'WasPassword' is not specified correctly.
   [logmsg] [2015-05-15 18:18:03.383] EJPCA1066E: PortalAdminCheckTask: Task FAILED. User
cannot authenticate with the specified password.
     [echo] ExceptionMessage is: Exception found when executing wsadmin
     [echo] StackTrace is:
com.ibm.wplc.deploy.aawsi.ant.WsadminSessionTask.execute(WsadminSessionTask.java:555)
     [echo] org.apache.tools.ant.UnknownElement.execute(UnknownElement.java:275)
     [echo] org.apache.tools.ant.Task.perform(Task.java:364)
     [echo] com.ibm.wps.config.TryCatchTask.execute(TryCatchTask.java:118)
===
```

Solution:
For a clustered environment, ensure the Deployment Manager is started. Then rerun *deploy-paa*.
For a standalone environment, *deploy-paa* will start the Portal server when needed.

Problem: Running *deploy-paa* fails on a predeployed application

In ConfigTrace.log:

```
===
Target started: action-install-content-xmlaccess-applySIFeaturePack-
components/WebSpherePortal.shared

[xmlaccess] EJPXB0006I: Connecting to URL http://localhost:10039/web/config/
[xmlaccess] EJPXB0002I: Reading input file
/opt/IBM/WebSphere/wp_profile/ConfigEngine/config/work/components/WebSpherePortal.shared/cont
ent/staging-version1.xml

[...]

[xmlaccess]    <status element="[web-app Z1_OIHA1402K04CD0AQ06ITT83001
uid=com.ibm.web.portlet.mycustomPortlet.8391422f94.webmod]" result="failed">
[xmlaccess]        <message id="EJPXA0183E">EJPXA0183E: The URL
file://localhost//opt/IBM/WebSphere/wp_profile/installedApps/targetCell/mycustomEAR.ear/mycusto
mWAR.war does not point to a directory as required by predeployed portlets. Please adjust the
URL.</message>
[xmlaccess]        <stacktrace><![CDATA[
[xmlaccess] com.ibm.wps.command.xml.XmlFormatException: EJPXA0183E: The URL
file://localhost//opt/IBM/WebSphere/wp_profile/installedApps/targetCell/mycustomEAR.ear/myc
ustomWAR.war does not point to a directory as required by predeployed portlets. Please adjust
the URL. [web-app Z1_OIHA1402K04CD0AQ06ITT83001
uid=com.ibm.web.portlet.mycustomPortlet.8391422f94.webmod]
[xmlaccess]  at
com.ibm.wps.command.xml.items.WebApplicationItem.init(WebApplicationItem.java:324)
[xmlaccess]  at com.ibm.wps.command.xml.DecodeEngine.makeItem(DecodeEngine.java:569)
[xmlaccess]  at com.ibm.wps.command.xml.DecodeEngine.processNodeSax(DecodeEngine.java:500)
[xmlaccess]  at
com.ibm.wps.command.xml.DecodeEngine.decodeTopLevelNode(DecodeEngine.java:146)
[xmlaccess]  at
com.ibm.wps.command.xml.XmlAccessHandler.fillObjectModel(XmlAccessHandler.java:322)
[xmlaccess]  at
com.ibm.wps.command.xml.XmlAccessHandler.endElement(XmlAccessHandler.java:212)
===
```

Solution:
The problem is corrected in APAR PI39049, which is included in CF06.
Apply the latest CF to obtain this APAR. Refer to the Recommended Updates.

Manual workaround:
The problem occurs because the pre-deployed EAR was extracted to a directory which specifies the source system's cell name.

The <url> tag can be manually updated. Review this XML file for any <url> tags that reference the $predeployed_root$ variable.

<wp_profile_root>/paa/WebSpherePortal/components/WebSpherePortal.shared/content/xmlaccess/install/staging-version1.xml

The same XML file can be updated in the PAA for other deployments.

WebSpherePortal.paa/WebSpherePortal/components/WebSpherePortal.shared/content/xmlaccess/install/staging-version1.xml

Problem: Hung thread in SystemOut log during *deploy-paa* when importing XML file through XMLAccess.

In the ConfigTrace.log, you may see an XMLAccess request take a long time, or appear to be hung.

```
===
[2015-05-15 18:32:37.785] Target finished: si-wait-for-sync-to-complete
    [copy] Copying 1 file to
/opt/IBM/WebSphere/wp_profile/ConfigEngine/config/work/components/WebSpherePortal.shared/content
[xmlaccess] EJPXB0006I: Connecting to URL http://localhost:10039/web/config/
[xmlaccess] EJPXB0002I: Reading input file
/opt/IBM/WebSphere/wp_profile/ConfigEngine/config/work/components/WebSpherePortal.shared/content/staging-version1.xml
===
```

In the SystemOut log, the following hung thread is seen:

```
[5/14/15 16:01:07:022 EDT] 000000b5 ThreadMonitor W   CWWSR0605W: Thread "WebContainer : 3" (000000ff) has been active for 702067 milliseconds and may be hung.  There is/are 1 thread(s) in total in the server that may be hung.
        at java.lang.Object.wait(Native Method)
        at java.lang.Object.wait(Object.java:196)
        at com.ibm.wps.pe.mgr.appserveradmin.WAS5Admin.startImpl(WAS5Admin.java:324)
        at com.ibm.wps.pe.mgr.appserveradmin.WAS5Admin.access$100(WAS5Admin.java:82)
        at com.ibm.wps.pe.mgr.appserveradmin.WAS5Admin$2.run(WAS5Admin.java:237)
        at com.ibm.ws.security.auth.ContextManagerImpl.runAs(ContextManagerImpl.java:5474)
        at
com.ibm.ws.security.auth.ContextManagerImpl.runAsSystem(ContextManagerImpl.java:5600)
        at com.ibm.wps.pe.mgr.appserveradmin.WAS5Admin.start(WAS5Admin.java:234)
        at
com.ibm.wps.pe.mgr.AbstractApplicationManagerImpl.setWebModuleState(AbstractApplicationManagerImpl.java:255)
        at
com.ibm.wps.pe.mgr.AbstractApplicationManagerImpl.startWebModule(AbstractApplicationManagerImpl.java:206)
        at
com.ibm.wps.command.applications.StartWebModuleCommand.execute(StartWebModuleCommand.java:137)
        at
com.ibm.wps.command.xml.items.WebApplicationItem.postProcess(WebApplicationItem.java:906)
        at com.ibm.wps.command.xml.UpdateEngine.postProcessItem(UpdateEngine.java:245)
        at com.ibm.wps.command.xml.UpdateEngine.processItem(UpdateEngine.java:227)
        at com.ibm.wps.command.xml.UpdateEngine.updateTopLevelItem(UpdateEngine.java:171)
        at
com.ibm.wps.command.xml.XmlAccessHandler.fillObjectModel(XmlAccessHandler.java:330)
        at com.ibm.wps.command.xml.XmlAccessHandler.endElement(XmlAccessHandler.java:212)
```

Solution:
This message is expected. The XMLAccess request will take a while to deploy and restart all of the applications. You can tail the SystemOut log to verify applications are still being restarted. You should continue to see messages like this:

```
[5/14/15 16:22:25:645 EDT] 00000245 ApplicationMg A   CWWSR0200I: Starting application:
PA_Impersonation
```

Problem: Warning messages about synchronization during execution of *deploy-paa*

In the ConfigTrace.log:

```
===
   [echo] EJPCE0070I: auto sync appears to be turned off, therefore, After deployment has completed, you will need to perform a manual sync of all nodes in the cluster
[2015-05-15 18:32:37.785] Target finished: si-wait-for-sync-to-complete
    [copy] Copying 1 file to /opt/IBM/WebSphere/wp_profile/ConfigEngine/config/work/components/WebSpherePortal.shared/content
[xmlaccess] EJPXB0006I: Connecting to URL http://localhost:10039/web/config/
[xmlaccess] EJPXB0002I: Reading input file /opt/IBM/WebSphere/wp_profile/ConfigEngine/config/work/components/WebSpherePortal.shared/content/staging-version1.xml
[xmlaccess] <?xml version="1.0" encoding="UTF-8"?>
[xmlaccess] <!-- IBM WebSphere Portal/8.5 build 20150507-2120 exported on Fri May 15 21:44:36 EDT 2015 from targetPortal85/9.37.21.11 virtual portal: none (default virtual portal) -->
[xmlaccess] <request build="20150507-2120" type="update" version="8.5.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="PortalConfig_8.5.0.xsd">
[xmlaccess]     <status element="[web-app Z1_00000000000000A0BR2B300IG4 uid=login.war.webmod]" result="warning">
[xmlaccess]         <message id="EJPXA0161W">EJPXA0161W: The web module Login Portlet Application could not be activated. Please see previous messages for reasons and possible corrective actions.</message>
[xmlaccess]     </status>
[xmlaccess]     <status element="[web-app Z1_00000000000000A0BR2B300IG4 uid=login.war.webmod]" result="warning">
[xmlaccess]         <message id="EJPPH0049W">EJPPH0049W: Automatic node synchronization is enabled for node [targetPortalNode, targetPortalNode, secondtargetPortal85Node]. However the portlet application PA_Login_Portlet_App will not be started in the Application Server. Manually start the application after all nodes were synchronized.</message>
[xmlaccess]     </status>
===
```

Solution:
Be sure automatic synchronization is turned on for the primary node in the cluster. This may have been disabled as part of a CF upgrade. Leave automatic synchronization turned off on secondary nodes until the PAA has been deployed the secondary node.

To re-enable, follow these steps:

1. In the administrative console for the Deployment Manager, select **System administration > Node agents** in the navigation tree.
2. Click **nodeagent** for the primary node.
3. Click **File Synchronization Service**.
4. Check the **Automatic Synchronization** check box to enable the automatic synchronization feature and check the **Enable service at server startup** check box to enable the synchronization service at server startup on the File Synchronization Service page and then click **OK**.
5. Click **Save** to save the configuration changes to the master repository.
6. Select **System administration > Nodes** in the navigation tree
7. Select the primary node and click on **Full Resynchronize**. The nodeagent will need to be started for this to work.
8. Select **System administration > Node agents** in the navigation tree
9. Select node agents for the primary node and click **Restart**.

Problem: Running *deploy-paa* fails on *import-wcm-data* with IllegalStateException

In the ConfigTrace.log:

```
Target started: run-wcm-admin-task

    [echo] VirtualPortalContext =  , VirtualPortalHost =
    [echo] Initialized: host targetPortal85  vpContext
    [echo] To call ConfigModuleClient: host targetPortal85  vpContext
[ConfigModuleClient] Connecting to login address:
http://targetPortal85.rtp.raleigh.ibm.com:10039/web/wcm/login
[ConfigModuleClient] Connecting to task address:
http://targetPortal85.rtp.raleigh.ibm.com:10039/web/wcm/myconnect?
MOD=data&processLibraries=false&taskType=import&input.dir=%2Fopt%2FIBM%2FWebSphere
%2Fwp_profile%2Fpaa%2FWebSpherePortal%2Fcomponents%2FWebSpherePortal.unique
%2Fcontent%2Fwcm%2FallLibraries&importLibrary=%24%7Bimport.library
%7D&skipScheduleActions=false with user name: wpsadmin
[ConfigModuleClient] No output will be shown while this task is running. Check the SystemOut.log
for progress information, or the z/OS Portal Job log.
[ConfigModuleClient] <html><head><title>IBM Web Content Manager Utility:
DataModule</title></head><body><h2>Log for Utility: DataModule</h2><p></p>Starting utility:
DataModule<br/>
[ConfigModuleClient] <font color=red>Check PortalServer log files for the detailed log of the module
from 3/6/15 10:55 PM</font><br/>Preserve dates option: False<br/>
[ConfigModuleClient] Report only mode <br/>
[ConfigModuleClient] Force unlock option: True<br/>
[ConfigModuleClient] The directory
"/opt/IBM/WebSphere/wp_profile/paa/WebSpherePortal/components/WebSpherePortal.unique/content/
wcm/allLibraries" was imported successfully.<br/>
[ConfigModuleClient] Failed to complete: com.aptrix.pluto.util.RepositoryException: Message:
Message: No message, Cause: java.lang.IllegalStateException: No controllable entry for library
{LibraryEntryImpl id:310b41e7-99f2-4ca7-b5c8-5be7b914498d, name: web content import 2, vp:
VirtualPortalEntry [vpId=0]}, Cause: javax.jcr.RepositoryException: Message: No message, Cause:
java.lang.IllegalStateException: No controllable entry for library {LibraryEntryImpl
id:310b41e7-99f2-4ca7-b5c8-5be7b914498d, name: web content import 2, vp: VirtualPortalEntry
[vpId=0]}<br/>
</body>
--- Exception Thrown ---
/opt/IBM/WebSphere/PortalServer/wcm/prereq.wcm/config/includes/prereq.wcm_cfg.xml:4034: An
exception occurred while executing the task: Status returned 500: Internal server error
        at
com.ibm.workplace.wcm.maintenance.module.ConfigModuleClient.execute(ConfigModuleClient.java:
103)
        at org.apache.tools.ant.UnknownElement.execute(UnknownElement.java:275)
        at org.apache.tools.ant.Task.perform(Task.java:364)
```

At the corresponding time in the SystemOut logs:

```
[3/6/15 23:37:32:667 CST] 000000f1 PlutoUtilityM E   Failed to complete: com.aptrix.pluto.util.RepositoryException: Message: Message: No message, Cause: java.lang.IllegalStateException: No controllable entry for library {LibraryEntryImpl id:310b41e7-99f2-4ca7-b5c8-5be7b914498d, name: web content import 2, vp: VirtualPortalEntry [vpId=0]}, Cause: javax.jcr.RepositoryException: Message: No message, Cause: java.lang.IllegalStateException: No controllable entry for library {LibraryEntryImpl id:310b41e7-99f2-4ca7-b5c8-5be7b914498d, name: web content import 2, vp: VirtualPortalEntry [vpId=0]}
                                 com.aptrix.pluto.util.RepositoryException: Message: Message: No message, Cause: java.lang.IllegalStateException: No controllable entry for library {LibraryEntryImpl id:310b41e7-99f2-4ca7-b5c8-5be7b914498d, name: web content import 2, vp: VirtualPortalEntry [vpId=0]}, Cause: javax.jcr.RepositoryException: Message: No message, Cause: java.lang.IllegalStateException: No controllable entry for library {LibraryEntryImpl id:310b41e7-99f2-4ca7-b5c8-5be7b914498d, name: web content import 2, vp: VirtualPortalEntry [vpId=0]}
	at com.ibm.workplace.wcm.data.module.DataModule.processUtility(DataModule.java:111)
	at com.aptrix.pluto.util.PlutoUtilityModule.processCurrentContext(PlutoUtilityModule.java:506)
	at com.aptrix.pluto.util.PlutoUtilityModule.process(PlutoUtilityModule.java:402)
	at com.presence.connect.business.module.ModuleManager.launchBusiness(ModuleManager.java:162)
	at com.presence.connect.business.module.ModuleManager.launchBusiness(ModuleManager.java:385)
	at com.presence.connect.RequestExecutable.execute(RequestExecutable.java:96)
	at com.presence.connect.dispatcher.Task.runWork(Task.java:151)
	at com.ibm.workplace.wcm.services.workmanager.AbstractWcmWork.runImpl(AbstractWcmWork.java:150)
	at com.ibm.workplace.wcm.services.workmanager.AbstractWcmWork.run(AbstractWcmWork.java:134)
	at com.presence.connect.ConnectClient.processSynchronous(ConnectClient.java:168)
	at com.presence.connect.ConnectServlet.process(ConnectServlet.java:308)
	at com.presence.connect.ConnectServlet.doGet(ConnectServlet.java:127)
```

An ffdc log may also be opened containing details of the stack trace.

Solution:
The problem is corrected in APAR PI36930, which is included in CF06.
Apply the latest CF to obtain this APAR. Refer to the Recommended Updates.

Problem: After running *deploy-paa* successfully, Portal does not render properly in a browser.

There is no theme rendered, and you may see this error in the browser:

EJPFD0097E: No theme was found to render the page. Navigate to Administration and assign a working theme to restore full function to your site.

In the SystemOut logs:

```
[5/17/15 16:43:06:509 EDT] 000000fe ServletWrappe E
com.ibm.ws.webcontainer.servlet.ServletWrapper service CWSRV0014E: Uncaught service()
exception root cause /themes/html/Default.jsp: com.ibm.websphere.servlet.error.ServletErrorReport:
com.ibm.wps.resolver.data.exceptions.URIProcessingIOException: EJCBD0021E: The URI
[war:custom8001Static/themes/custom8001Theme/theme.html] and parameters
[[&#39;themeURI&#39;=[&#39;war:custom8001Static/themes/custom8001Theme/theme.html&#39;],
&#39;mime-type&#39;=[&#39;text/html&#39;]]] could not be processed: [EJCBD0021E: The URI
[war:custom8001Static/themes/custom8001Theme/nls/theme_en.html] and parameters
[[&#39;themeURI&#39;=[&#39;war:custom8001Static/themes/custom8001Theme/theme.html&#39;],
&#39;mime-type&#39;=[&#39;text/html&#39;]]] could not be processed: [EJCBD0021E: The URI
[res:/web/themeModules/modules/pagebuilder/jsp/preview.jsp] and parameters []] could not be
processed: [EJCBC0009E: The entity [/web/themeModules/modules/pagebuilder/jsp/preview.jsp] could
not be be found.]]]
	at org.apache.jasper.runtime.PageContextImpl.handlePageException(PageContextImpl.java:693)
	at com.ibm._jsp._Default._jspService(_Default.java:1159)
	at com.ibm.ws.jsp.runtime.HttpJspBase.service(HttpJspBase.java:99)
	at javax.servlet.http.HttpServlet.service(HttpServlet.java:668)
```

Caused by: com.ibm.wps.resolver.data.exceptions.URIProcessingIOException: EJCBD0021E: The URI [res:/web/themeModules/modules/pagebuilder/jsp/preview.jsp] and parameters []] could not be processed: [EJCBC0009E: The entity [/web/themeModules/modules/pagebuilder/jsp/preview.jsp] could not be be found.]

```
	at com.ibm.wps.resolver.resource.ResourceDataSourceFactory.newSource(ResourceDataSourceFactory.java:957)
	at com.ibm.portal.resolver.helper.CORDataSourceFactoryEx.newSource(CORDataSourceFactoryEx.java:50)
	at com.ibm.wps.resolver.contentspot.ContentSpotDataSourceFactory.newSource(ContentSpotDataSourceFactory.java:194)
```

```
Caused by: com.ibm.wps.resolver.resource.exceptions.ServletContextNotFoundException: EJCBC0009E: The entity [/web/themeModules/modules/pagebuilder/jsp/preview.jsp] could not be be found.
        at com.ibm.wps.resolver.resource.ResourceDataSourceFactory.createRequestDispatcherDataSource(ResourceDataSourceFactory.java:698)
        at com.ibm.wps.resolver.resource.ResourceDataSourceFactory.newSource(ResourceDataSourceFactory.java:917)
        ... 108 more
```

Solution:
Ensure the context root of the target environment matches the same context root used on the source system. Run *modify-servlet-path* as described in the knowledge center:

http://www-01.ibm.com/support/knowledgecenter/SSHRKX_8.5.0/mp/config/cfg_intr.dita?lang=en

Acknowledgements

The IBM WebSphere Portal and IBM Web Content Manager support teams collaborated to produce this white paper.

David Batres supports WebSphere Portal security/administration and authored this white paper
David Nixon develops and architects WebSphere Portal design and provided technical assistance and guidance
Vasanth Alle supports WebSphere Portal security/administration and assisted with security configurations and testing
Jason Wicker supports WebSphere Portal security/administration and provided technical assistance and guidance
Travis Cornwell supports WebSphere Portal security/administration and provided technical assistance and guidance
Ben Parker supports WebSphere Portal security/administration and assisted with installation, database configuration, and clustering
Michael Watkin supports WebSphere Portal installation/configuration and assisted with installation, database configuration, and clustering
Oscar Hernandez supports WebSphere Portal API/theme development and assisted with custom applications and theme deployment
Dirk Tysmans supports WebSphere Portal API/theme development and assisted with custom applications and theme deployment
Jack Derbyshire supports Web Content Management and assisted with WCM content and personalization rules

References

[1]Overview of staging to production
http://www-01.ibm.com/support/knowledgecenter/SSHRKX_8.5.0/mp/deploy/dep_ovr_concepts.dita?lang=en

[2]Staging to production process
http://www-01.ibm.com/support/knowledgecenter/SSHRKX_8.5.0/mp/deploy/dep_intr.dita?lang=en

[3]PAA file structure overview
http://www-01.ibm.com/support/knowledgecenter/SSHRKX_8.5.0/mp/config/si_paa_spec_ov.dita?lang=en

[4]Components directory
http://www-01.ibm.com/support/knowledgecenter/SSHRKX_8.5.0/mp/config/si_paa_spec_compdir.dita?lang=en

[5]Preparing the servers for initial staging
http://www-01.ibm.com/support/knowledgecenter/SSHRKX_8.5.0/mp/deploy/dep_prep.dita?lang=en

[6]Syndication and staging
http://www-01.ibm.com/support/knowledgecenter/SSHRKX_8.5.0/mp/deploy/dep_up_syn.dita?lang=en

Developing basic PAA file applications
http://www-01.ibm.com/support/knowledgecenter/SSHRKX_8.5.0/mp/config/dev_sol_app.dita?lang=en

Developing advanced PAA file applications
http://www-01.ibm.com/support/knowledgecenter/SSHRKX_8.5.0/mp/config/dev_sol_app_adv.dita?lang=en

Syndication
http://www-01.ibm.com/support/knowledgecenter/SSHRKX_8.5.0/mp/wcm/wcm_administering.dita?lang=en

dWAnswers
https://developer.ibm.com/answers/topics/digexp/

Portal Log Analyzer
https://greenhouse.lotus.com/plugins/plugincatalog.nsf/assetDetails.xsp?action=editDocument&documentId=C7AB4C82F17EC6228525790B006F6D43

Change History

Version	Date	What Changed
1.0	10/16/15	Published