# **AWS-S3 Storage Extension** – **Deployment Process on CD Server**

Implementation requires the following steps:

1. **Update the Deployer service**
[These steps must be repeated for all (staging and live) Deployer service in a given environment]
* Create the **aws-s3** folder into the services folder of the Deployer Service i.e. in CD Server (for all staging and Live Environment)

Note: this folder contains the Custom aws s3 binary/Page/CP storage extension and dependencies

Create a Folder with Name **aws-s3** under the Service Folder of Deployer Service (Login to CD Server)


Place the below **Jar** files + the **Storage Extension Jar** file i.e. aws-s3-1.1-SNAPSHOT.jar as well



The below are the DAO and Factory Files for each typeMapping:

|  |  |  |
| --- | --- | --- |
| **typeMapping** | **DAO Files** | **DAO Factory Files** |
| Binary | JPAAWSBinaryContentDAO | JPAAWSDAOFactory |
| Pages | JPAAWSPageDAO | JPAAWSPageDAOFactory |
| ComponentPresentation | JPAAWSComponentPresentationDAO | JPAAWSCPDAOFactory |

**Note: All the DAO Files Configuration goes to AWSDAOBundle.xml File**

* Copy the **AWSDAOBundle.xml** file into the **config** folder of the Deployer Service



**Note: All the “DAOFactory” Files Config’s goes to cd\_storage\_config.xml**

* In the file **cd\_storage\_conf.xml**, located in the config folder of the Deployer Service

Add <**Bundle src="AWSDAOBundle.xml"**/> right before the closing tag

For each itemMapping we need add respective DAOFactory Class and where it needs to get Store i.e. either filesystem or DB, Need to provide the respective ServerName, PortNo, DBName, User and Pwd based on the Environments

Add for Binary

<Storage Class="com.tridion.storage.aws.JPAAWSDAOFactory" Id="AWSS3Binary" Type="persistence" dialect="MSSQL">

 <Pool CheckoutTimeout="120" IdleTimeout="120" MonitorInterval="60" Size="5" Type="jdbc"/>

 <DataSource Class="com.microsoft.sqlserver.jdbc.SQLServerDataSource">

 <Property Name="serverName" Value="localhost"/>

 <Property Name="portNumber" Value="1433"/>

 <Property Name="databaseName" Value="Broker\_Stg"/>

 <Property Name="user" Value="Broker\_Stg\_User"/>

 <Property Name="password" Value="xxxxxxxx"/>

 </DataSource>

 <AWSS3BinaryIndexer AwsAccessKeyId="XXXXX" AwsSecretAccessKeyId="XXXX" BucketName="XXXXX" Class="com.tridion.storage.aws.AWSS3BinaryIndexer"/>

</Storage>

Add for Page

<Storage Class="com.tridion.storage.aws.JPAAWSPageDAOFactory" Id="AWSS3Page" Type="persistence" dialect="MSSQL">

 <Pool CheckoutTimeout="120" IdleTimeout="120" MonitorInterval="60" Size="5" Type="jdbc"/>

 <DataSource Class="com.microsoft.sqlserver.jdbc.SQLServerDataSource">

 <Property Name="serverName" Value="localhost"/>

 <Property Name="portNumber" Value="1433"/>

 <Property Name="databaseName" Value="Broker\_Stg"/>

 <Property Name="user" Value="Broker\_Stg\_User"/>

 <Property Name="password" Value="xxxxxxxx"/>

 </DataSource>

 <AWSS3PageIndexer AwsAccessKeyId="XXXXX" AwsSecretAccessKeyId="XXXX" BucketName="XXXXX" Class="com.tridion.storage.aws.AWSS3PageIndexer"/>

</Storage>

Add for ComponentPresentation

<Storage Class="com.tridion.storage.aws.JPAAWSCPDAOFactory" Id="AWSS3CP" Type="persistence" dialect="MSSQL">

 <Pool CheckoutTimeout="120" IdleTimeout="120" MonitorInterval="60" Size="5" Type="jdbc"/>

 <DataSource Class="com.microsoft.sqlserver.jdbc.SQLServerDataSource">

 <Property Name="serverName" Value="localhost"/>

 <Property Name="portNumber" Value="1433"/>

 <Property Name="databaseName" Value="Broker\_Stg"/>

 <Property Name="user" Value="Broker\_Stg\_User"/>

 <Property Name="password" Value="xxxxxxxx"/>

 </DataSource>

 <AWSS3CPIndexer AwsAccessKeyId="XXXXX" AwsSecretAccessKeyId="XXXX" BucketName="XXXXX" Class="com.tridion.storage.aws.AWSS3CPIndexer"/>

</Storage>

Right before the closing tag **</Storages>**

 **Note:** Update the correct database configuration details respective environment and AWS S3 bucket and API details

* Add

<Item cached="true" itemExtension=".aspx" storageId="defaultdb" typeMapping="Page"/>

<Item cached="false" storageId="AWSS3Binary" typeMapping="Binary"/>

<Item cached="false" storageId="AWSS3Page" typeMapping="Page"/>

Right before the closing tag </**ItemTypes**>

Just for Reference am attaching the current **cd\_storage\_config.xml** file

 

* To enable logging, it is required to add a logger into **logback.xml**, if this is not already available

   <logger name="com.tridion.storage.aws" level="${log.level}" additivity="false">

 <appender-ref ref="rollingAWSExtensionsLog" />

 </logger>

 **Note:** by replacing “${log.level}” with “**DEBUG**” for this appender only, the loglevel can be set for the aws-s3 Extensions, without influencing the other logging.

This logger is configured to use a separate log file, which is defined by adding the appender below

   <appender name="rollingAWSExtensionsLog" class="ch.qos.logback.core.rolling.RollingFileAppender">

 <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">

 <fileNamePattern>${log.folder}/awss3-extentions.%d{yyyy-MM-dd}.log</fileNamePattern>

 <maxHistory>${log.history}</maxHistory>

 </rollingPolicy>

 <encoder>

 <charset>${log.encoding}</charset>

 <pattern>${log.pattern}</pattern>

 </encoder>

 <prudent>true</prudent>

 </appender>

**Note 1:** If logging into a separate file is not required, the appender does not have to be added and the logger will look like this

    <logger name=" com.tridion.storage.aws" level="${log.level}">

        <appender-ref ref="rollingCoreLog"/>

    </logger>

 **Note 2:** by replacing “${log.level}” with “**DEBUG**” for this appender only, the loglevel can be set for the aws-s3 Extensions, without influencing the other logging.

It is strongly recommended to set log level higher to **DEBUG** unless debugging is being performed.

* + Re-Install the Deployer Service to pick the **aws-s3** folder and jars
	+ Restart the Deployer Micro service
	+ Check the **cd\_core** log file for errors if any
1. **Republish the pages which has images, .xml’s, DCP ,.aspx or any other Binaries**
2. **Verify the results in aws s3 bucket and consult awss3-extentions log file by debug mode if required**