

Back Up and Restore WordPress

Contents

[Introduction](#)

[Prerequisites](#)

[Requirements](#)

[Components Used](#)

[Background Information](#)

[Gather Required Information](#)

[Download and Update Scripts](#)

[Update WordPress Profile](#)

Introduction

This document describes how to use CloudCenter to migrate an application to back up and restore the content to an Amazon S3 Bucket.

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- Linux
- CloudCenter
- Amazon S3

Components Used

The information in this document is based on CloudCenter v4.8.1.1.

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, make sure that you understand the potential impact of any command.

Background Information

This document assumes that CloudCenter is already installed and works properly. Also, WordPress has already been modeled and confirmed to work properly. This process has only been tested with the use of an S3 Bucket as the repository and migrates from one deployment to another between public clouds, though it must work between public and private clouds as long as

connectivity with the S3 Bucket is confirmed in the destination cloud. This procedure must only be done for a proof of concept. It does not utilize best practice as secret keys are hard coded into the script for ease of use. The scripts specified were written for CentOS with the use of an Apache webserver.

Gather Required Information

CloudCenter uses a few scripts to be able to backup and restore the data to the correct VM, in order to populate the scripts some information is needed to be gathered beforehand.

- Name of the Amazon S3 Bucket
- Path within the bucket where backup data is stored
- Amazon S3 Key
- Amazon S3 Secret Key

Note: S3 bucket must be added as a repository in CloudCenter as well.

Download and Update Scripts

- Four Scripts are necessary to handle migrating WordPress. Two backup scripts, one for the database and another for the webserver, and two restore scripts.
- Download **BackupRestore.zip** and extract from it **RestoreServer.sh**, **RestoreDB.sh**, **BackupDB.sh** and **BackupServer.sh**.
- From within each one, update the **Bucket**, **Path**, **S3 Key**, **S3 Secret**.
- The backup script for the webserver zips up the **/var/www/** directory into a file called **server.zip** which is stored in the **/tmp** directory. It then uploads the **server.zip** to the S3 Bucket with the credentials specified.
- The restore script for the webserver, downloads the **server.zip** file and unzips it into the **/var/www/** directory. Neither of these scripts do any error checks, nor do they check the OS installed, this can cause issues if WordPress was installed on a different OS or with a different webserver, other than Apache.
- The backup script for the database performs a database (DB) dump and zips it up before it uploads to the S3 Bucket.
- The restore script for the database creates the DB and then uses the DB dump it downloaded from the S3 Bucket to recreate the database.

Note: These scripts have the S3 Key and Secret stored in plain text, this is not recommended and must only be used as a Proof Of Concept, or at the time of initial testing.

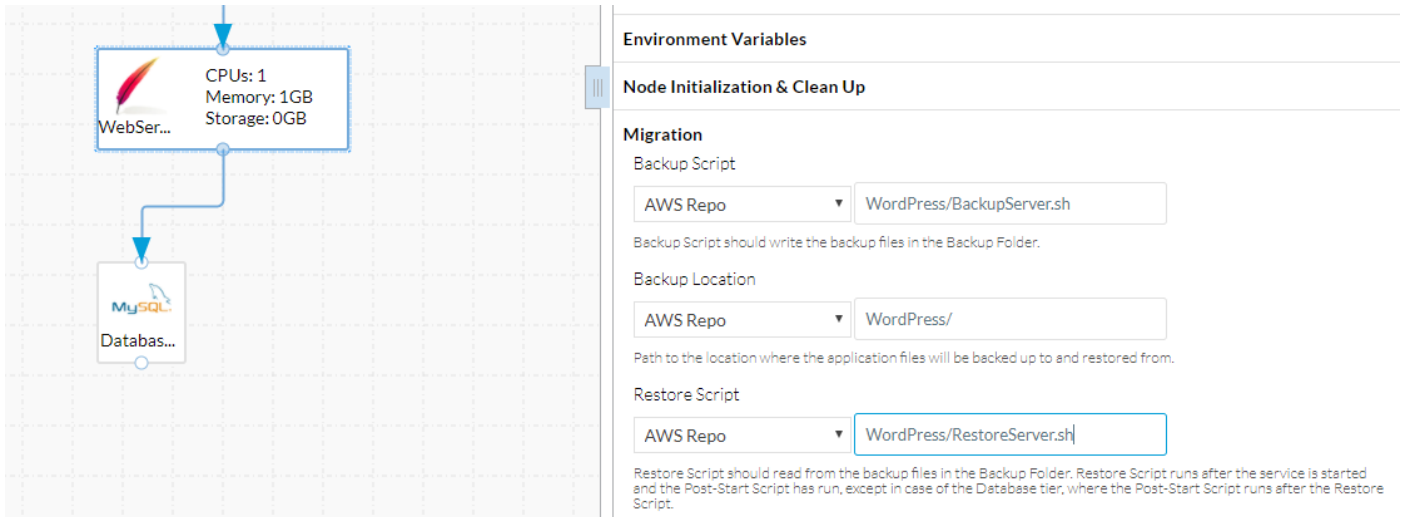
After all fields have been updated, upload the scripts to a CloudCenter repository so that they can be referenced within an Application Profile.

Update WordPress Profile

A few updates to the profile is needed to be made to make use of these new scripts.

Under the **WebServer**, select **Migration** and add a path to the **BackupServer.sh** in **Backup**

Script, also reference the Backup Location in the **Backup Location**, and finally add the path to the **RestoreServer.sh** in **Restore Script** as shown in the image.

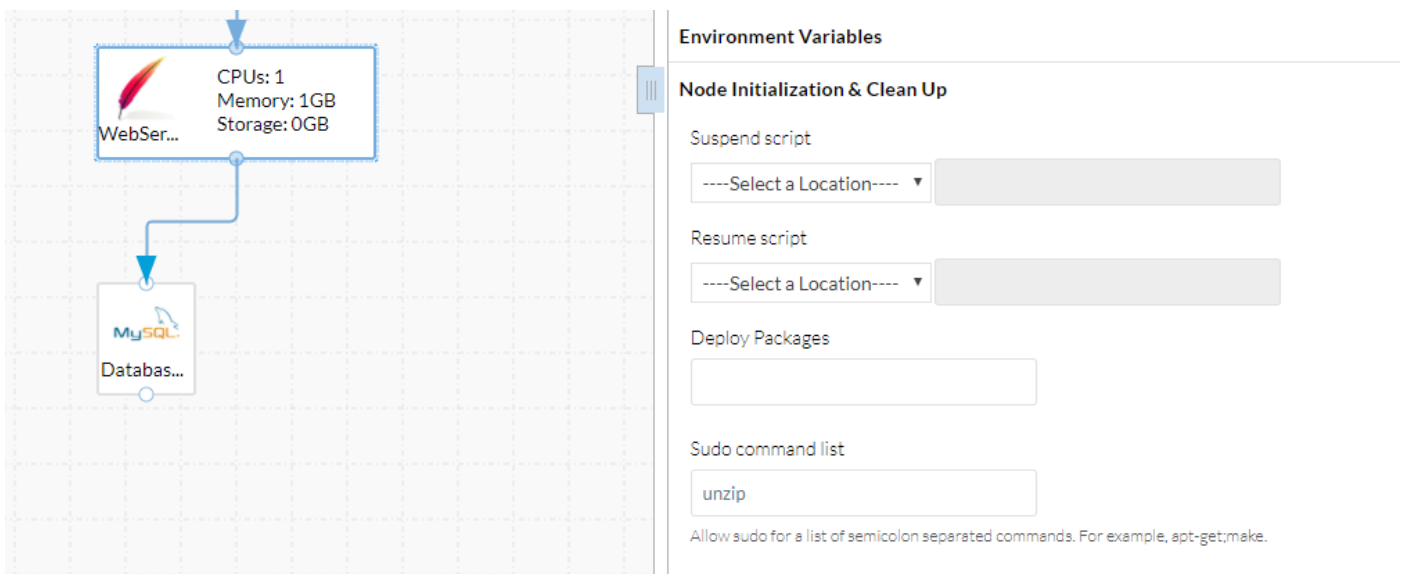


The screenshot shows a cloud management console interface. On the left, a diagram illustrates a two-tier application stack. The top tier is a WebServer with 1 CPU, 1GB Memory, and 0GB Storage. The bottom tier is a Database. On the right, the configuration page is displayed. The 'Migration' section is expanded, showing the following settings:

- Backup Script:** WordPress/BackupServer.sh
- Backup Location:** WordPress/
- Restore Script:** WordPress/RestoreServer.sh

Below the Restore Script field, a note states: "Restore Script should read from the backup files in the Backup Folder. Restore Script runs after the service is started and the Post-Start Script has run, except in case of the Database tier, where the Post-Start Script runs after the Restore Script."

The RestoreServer.sh does need permission to be able to unzip the files to **/var/www/** which cliqruser does not have permission to do. Under **Node Initialization & Clean Up** add **unzip** to the Sudo command list. This gives the script the permission to run unzip as root as shown in the image.



The screenshot shows the same cloud management console interface. The 'Node Initialization & Clean Up' section is now expanded, showing the following settings:

- Suspend script:** ----Select a Location----
- Resume script:** ----Select a Location----
- Deploy Packages:** (empty field)
- Sudo command list:** unzip

Below the Sudo command list field, a note states: "Allow sudo for a list of semicolon separated commands. For example, apt-get;make."

The Database Tier needs similar changes as the WebServer, namely the **Backup Script**, **Backup Location**, and **Restore Script** as shown in the image.

The screenshot displays a configuration interface for an application profile. On the left, a diagram shows a 'WebSer...' component connected to a 'MySQL Databas...' component. The MySQL component is specified with 'CPU: 1', 'Memory: 1GB', and 'Storage: 0GB'. On the right, the 'Migration' section is configured with the following settings:

- Script executed before the backup is taken:** (Empty field)
- Backup Script:** AWS Repo | WordPress/BackupDB.sh
- Backup Script should write the backup files in the Backup Folder:** (Empty field)
- Backup Location:** AWS Repo | WordPress
- Path to the location where the application files will be backed up to and restored from:** (Empty field)
- Restore Script:** AWS Repo | WordPress/RestoreDB.sh
- Restore Script should read from the backup files in the Backup Folder. Restore Script runs after the service is started and the Post-Start Script has run, except in case of the Database tier, where the Post-Start Script runs after the Restore Script.** (Empty field)
- Post-Migrate Script:** (Empty field)

Once these changes are made simply **Save** the Application Profile.

Now a new deployment must be able to be migrated from one node to another.