

Upgrade an Application Using CloudCenter

Contents

[Introduction](#)

[Prerequisites](#)

[Requirements](#)

[Components Used](#)

[Background Information](#)

[Define Upgrade Process](#)

[Create New Version](#)

[Deploy Application](#)

Introduction

This document describes the process to upgrade an application using CloudCenter.

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- CloudCenter
- Bash

Components Used

The information in this document is based on CloudCenter 4.8.1.1.

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

There are multiple ways to upgrade an application in CloudCenter. One option is to create a custom action that can be applied to individual VM's and runs an upgrade script. This method gives you complete control over the upgrade and allows testing of one node before upgrading the next node. The downside is that it is a very manual process that requires writing individualized scripts for every upgrade. The preferred method is to make use of CloudCenter's upgrade framework to automate the upgrade process.

Define Upgrade Process

Edit "Upgrade Application" Application Profile

Version: [1.0](#) (Revision: 3) > [2.0](#)

Basic Information | Global Parameters | **Topology Modeler**

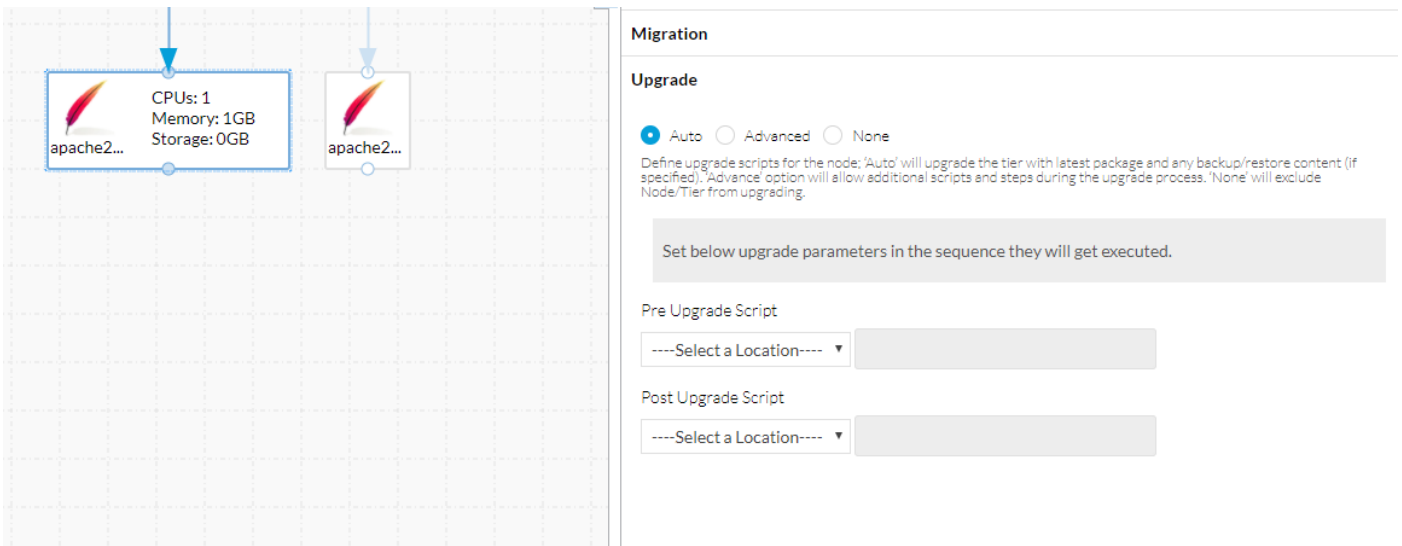
Services

- Message Bus
- OS Service
- Custom Service
- File System
- Workflow
- Orchestration
- Frontend Cache
- Load Balancer
- Web Server

```
graph TD; nginx_1[nginx_1] --> apache2_1[apache2...]; nginx_1 --> apache2_2[apache2...];
```

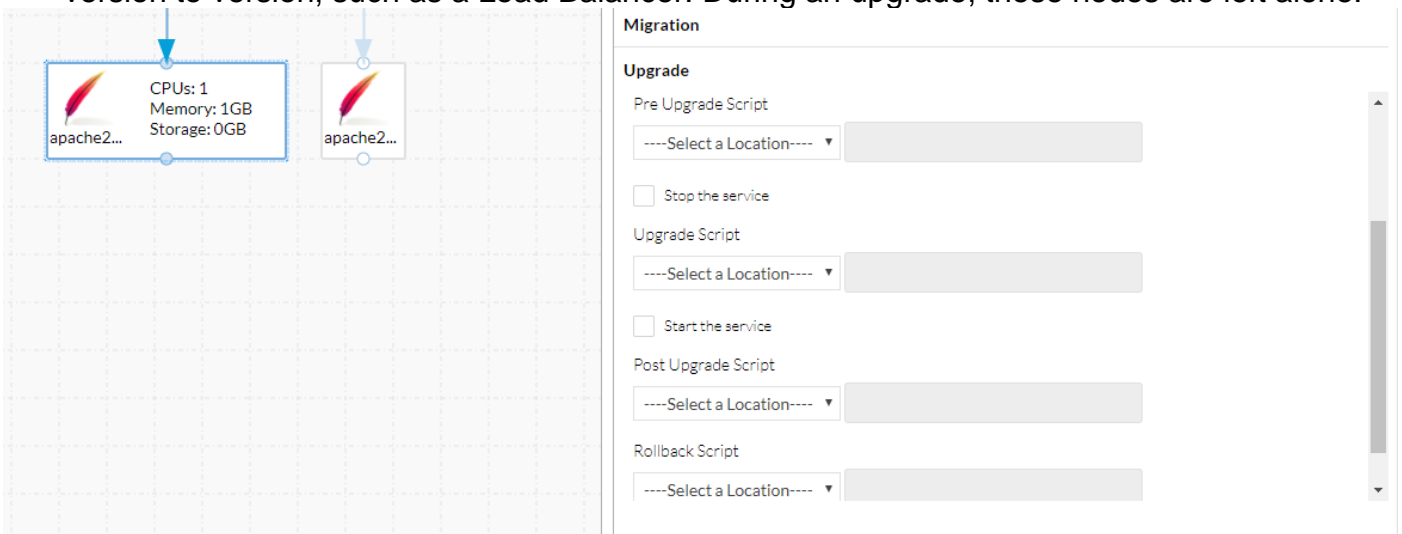
In this sample application, there are two Apache web servers behind a Nginx load balancer. These web servers are identical and provide HA availability to a website that is being hosted. An ideal upgrade process allows the nodes to be upgraded individually so that there is always a node hosting the website allowing for 100% uptime during the upgrade process.

By default, during an upgrade CloudCenter downloads any new packages and content, then make use of any backup and restore scripts to persist data. If more in-depth logic is needed, then upgrade scripts can be included.



Under **Migration** tab, the backup and restore scripts can be found. Those are used both for Migration and Upgrade. The **Upgrade** tab has three options: **Auto**, **Advanced**, **None**.

- Auto allows CloudCenter to automatically upgrade the node, it downloads the new content, and runs the backup and restores scripts to preserve important information.
- Advanced allows the complete control of the upgrade process.
- None means do not upgrade this node, it can be done for nodes that have no changes from version to version, such as a Load Balancer. During an upgrade, these nodes are left alone.



Advanced allows more scripts to be added and allows you to stop and start the service during the upgrade.

Once all necessary upgrade actions are defined, it is important to **save** the Application before moving on to the next step

Create New Version

After you save the application, navigate back to the **Topology Modeler**.

Edit "Upgrade Application" Application Profile

Version: [1.0](#) (Revision: 3) > [2.0](#)

Basic Information Global Parameters **Topology Modeler**

Services

- Message Bus
- OS Service
- Custom Service
- File System
- Workflow
- Orchestration
- Frontend Cache
- Load Balancer
- Web Server

Apache2
Open-source HTTP server for OS

Geronomo3
Open source application server

IIS
Web server for Windows-based apps

Jetty
Java-based HTTP server

The diagram shows a central node labeled 'nginx_1' with the NGINX logo. Two lines extend downwards from this node, each ending in an arrowhead pointing to a separate node labeled 'apache2...'. Each 'apache2...' node features the Apache logo. The entire diagram is set against a light gray grid background. At the top of the diagram area, there are three icons: a magnifying glass, a magnifying glass with a minus sign, and a circular arrow. A 'Clear' button is located in the top right corner of the diagram area.

CloudCenter handles upgrading with the help of versioning. The application in the picture above is at Version 1.0, this can be seen in the upper left corner. In order to make use of CloudCenter's upgrade tool, a new version must be made.

- Select **Basic Information**.
- Enter a new **Version**.

Edit "Upgrade Application" Application Profile

Version: [1.0](#) (Revision: 3) > [2.0](#)

Basic Information

Global Parameters

Topology Modeler

Web App Name *

Upgrade Application

Version *

2.0

Revision

3

CloudCenter saves Version 1.0 and puts all new changes in Version 2.0.

This tells CloudCenter that there is a new version, and allows it to track the differences. Since this application is just two web servers, the only difference is to update the **Application Package** to point to a new zip file.

The application can be saved again.

Deploy Application

Now, when you deploy the application, you can choose which version to deploy. For this example, the original version is deployed.

General Settings

* DEPLOYMENT NAME

UpgradeExample

* APPLICATION VERSION

2.0

1.0

2.0

Enter Tag Name

TERMINATE PROTECTION



AGING POLICY

Once the application is deployed it can be upgraded from the Deployments Screen.

Deployment Name	Status	Environment	Created At	Duration	Cost	Actions
UpgradeExample Upgrade Application (V1.0) AWS/us-east-1	Deployed	Dev	20 Dec 2017 at 08:56 AM	6 mins	\$0.04	-Actions- Suspend Terminate Terminate And Hide Upgrade Promote Migrate Enable Terminate Protection Share
queueManTest8 QueueMan (V2.0) AWS/us-east-1	Stopping	Dev	19 Dec 2017 at 02:33 PM	18 hrs 25 mins	\$0.23	
QueueManTest6 QueueMan (V2.0) AWS/us-east-1	Terminating	Dev	19 Dec 2017 at 02:05 PM	18 hrs 53 mins	\$0.23	
QueueManTest5 QueueMan (V2.0) AWS/us-east-1	Terminated	Dev	19 Dec 2017 at 01:55 PM	9 mins	\$0.01	
QueueManTest3						

The upgrade process starts from the lowest tier and happens one node at a time. For our two-tier application, one Apache web server is upgraded.

Once that is completed, the second is upgraded. If you have defined an upgrade process for the Nginx load balancer, it is upgraded in the last.