Cisco Media Transformer 1.0 Installation Guide

February 1, 2018

Cisco Systems, Inc.
www.cisco.com

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco website at www.cisco.com/go/offices.

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: www.cisco.com/go/trademarks. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1110R)

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

© 2018 Cisco Systems, Inc. All rights reserved.

Preface vii

CHAPTER 1 **Cisco Media Transformer Overview** 1-1

Product Overview 1-1
Containerized Deployment 1-1
Functional Overview 1-2
Worker Node Deployment 1-2
CMT Network Overview 1-3
LB Request Example 1-3
Virtual Machine Types 1-4
Master VMs 1-4
Deployer VMs 1-5
Load Balancer VMs 1-5
Infrastructure VMs 1-5
Worker VMs 1-6
System Hardware Requirements 1-7
Terms and Definitions 1-7

CHAPTER 2 **Installation Prerequisites** 2-1

Pre-installation Tasks 2-2
Configuring the UCS Servers 2-2
Configuring CIMC Access 2-2
Configuring Drives & Controllers 2-4
Configuring the CPU 2-6
Mapping Virtual Media 2-7
Installing ESXi 2-7
Configuring ESXi 2-8
Installing the Virtual Machines 2-8
Deploying OVA Images 2-8
Assigning IP Addresses 2-9
Configuring Swap Memory 2-11
Configuring NTP 2-12

CHAPTER 3 **Installation** 3-1

Editing the Inventory File 3-1
Increase Timeout for Docker Image Load 3-7
Update the dnsmasq 3-7
Verifying Node Accessibility 3-8
Running the Ansible Playbook 3-8
Performing the Installation 3-9
Verifying the Installation 3-13
OpenShift Verification Commands 3-13
Command Line Verification 3-13
Verifying the NIC & Node Labels 3-16
GUI Verification 3-16
Updating the Cluster Port Range 3-17
Updating iptables 3-20
Configuring the IPVS VIP on all Worker Nodes 3-20
Verifying the IPVS VIP on all Worker Nodes 3-25
Load Images into Docker Registry 3-25
Verifying Docker Image Loading 3-33
Create the ABR2TS Project Namespace 3-34
Configuring VoD Gateway & Fluentd Pods 3-35
Logging Queue Deployment 3-36
Configuring the Logging Queue 3-37
Deploying the Logging Queue to the Cluster 3-38
Starting VoD Gateway & Fluentd 3-42
Verifying VoD Gateway & Fluentd Startup 3-43
Stopping VoD Gateway & Fluentd 3-43
Configuring Splunk for use with CMT 3-44
Verifying Connectivity with Splunk 3-45
Configuring IPVS 3-45
Verifying Node Access 3-46
Starting IPVS 3-47
Verifying IPVS is Running 3-48
Determining where IPVS Master is Running 3-50
Stopping IPVS 3-50
Running the Ingress Controller Tool 3-51
Monitoring Stack Overview 3-54
Installing the Monitoring Stack 3-54
Starting the Monitoring Stack 3-58
Stopping the Monitoring Stack 3-59

Verifying the Cluster 3-60
Configuring Grafana 3-61
Importing Grafana Dashboards 3-62
Adding Routes for Infra & Worker Nodes 3-62

APPENDIX A **Ingesting & Streaming Content** A-1
Provisioning ABR Content A-1
Verifying Ingestion Status A-3
Streaming ABR Content A-4

APPENDIX B **Heapster Logs** B-1
Heapster Overview B-1
Aggregates B-1

APPENDIX C **Alert Rules** C-1
Alert Rules Overview C-1
Updating Alert Rules C-1
Alert Rules Reference Materials C-2
Sample Alert Rule C-2
Alert Rule Commands C-2
Inspecting Alerts at Runtime C-3
Sending Alert Notifications C-3
Sample Alert Notifications C-5

Preface

The following guide provides installation instructions and relevant theory for Cisco's Media Transformer (CMT) solution.

New and Changed Information

Given that this is a new product release, all information within this document is also new. Return to this section in future releases to determine what has changed.

Audience

This guide is intended for use by network administrators responsible for installing, configuring, and troubleshooting the CMT solution and related software components. We expect that the reader will already be familiar with Linux, OpenShift, Kubernetes, Docker, and containerized software in general. Additionally, an understanding of VOD, OTT, and Legacy TV network infrastructure is beneficial, though, in places, we will review relevant concepts within this guide.

Document Organization

This document contains the following chapters and appendices:

Chapter or Appendix	Description
Cisco Media Transformer Overview	Introduces the theory behind CMT along with key terminology and concepts. This chapter explains the containerized deployment model, provides a functional overview, and explains the various virtual machine types within the solution. Lastly, System Hardware Requirements are mentioned at a high level.

Chapter or Appendix	Description
Installation Prerequisites	This chapter covers pre-installation tasks that must be performed to prepare your servers for the primary installation of CMT. Tasks include the initial setup and configuration of UCS servers, installing and configuring ESXi, and configuring and installing the virtual machines.
Installation	This is the bulk of the installation guide. This chapter includes instructions for editing the inventory file and performing the installation. Additional sections cover topics such as loading images into the Docker registry, creating the project namespace, logging queue deployment, IPVS, and the Monitoring Stack. Verification steps are included after most procedures to ensure that the relevant software is correctly installed and configured.
Ingesting & Streaming Content	This appendix provides instructions and verification steps on ingesting and streaming ABR content.
Heapster Logs	This appendix provides information on the Heapster metric-gathering tool.
Alert Rules	This appendix explains Alert Rules and provides information and resources that can be used to customize the rules for your deployment.

Document Conventions

This document uses the following conventions:

Convention	Indication
bold font	Commands and keywords and user-entered text appear in **bold** font.
italic font	Document titles, new or emphasized terms, and arguments for which you supply values are in *italic* font.
[ ]	Elements in square brackets are optional.
{x | y | z }	Required alternative keywords are grouped in braces and separated by vertical bars.
[x | y | z]	Optional alternative keywords are grouped in brackets and separated by vertical bars.
string	A non-quoted set of characters. Do not use quotation marks around the string or the string will include the quotation marks.
`courier` font	Terminal sessions and information the system displays appear in `courier` font.
< >	Non-printing characters such as passwords are in angle brackets.
[ ]	Default responses to system prompts are in square brackets.
!, #	An exclamation point (!) or a pound sign (#) at the beginning of a line of code indicates a comment line.

Note Means *reader take note*. Notes contain helpful suggestions or references to material not covered in the manual.

Tip Means *the following information will help you solve a problem*. The tips information might not be troubleshooting or even an action, but could be useful information, similar to a Timesaver.

Caution Means *reader be careful*. In this situation, you might perform an action that could result in equipment damage or loss of data.

Timesaver Means *the described action saves time*. You can save time by performing the action described in the paragraph.

Warning **IMPORTANT SAFETY INSTRUCTIONS**

This warning symbol means danger. You are in a situation that could cause bodily injury. Before you work on any equipment, be aware of the hazards involved with electrical circuitry and be familiar with standard practices for preventing accidents. Use the statement number provided at the end of each warning to locate its translation in the translated safety warnings that accompanied this device.

SAVE THESE INSTRUCTIONS

Warning **Statements using this symbol are provided for additional information and to comply with regulatory and customer requirements.**

Related Publications

Refer to the following documents for additional information about CMT 1.0:

- *Release Notes for Cisco Media Transformer 1.0*
- *Cisco Media Transformer 1.0 User Guide*
- *Open Source used in Cisco Media Transformer 1.0*

Cisco Media Transformer Overview

This chapter includes the following topics to introduce you to the Cisco Media Transformer (CMT) solution:

- Product Overview, as shown below
- Containerized Deployment, page 1-1, as shown below
- Functional Overview, page 1-2
- CMT Network Overview, page 1-3
- Virtual Machine Types, page 1-4
- System Hardware Requirements, page 1-7
- Terms and Definitions, page 1-7

Product Overview

Cisco's Media Transformer (CMT) is a part of the OMD (Open Media Distribution) Suite of products. The CMT solution provides fill-agent functionality to VDS-TV VoD streamers and transforms MPEG DASH TS (segmented-ABR) content to MPEG-2 TS-compliant streams, which allows playback of ABR content on legacy set-top boxes that require CBR input. This approach effectively allows Service Providers to fully leverage their existing QAM-based set-top box infrastructure, while giving them a path to transition to IP-based set top boxes over a longer timeframe.

Note During the development stages, Cisco Media Transformer has undergone a name change from ABR2TS. That older acronym may still appear in configuration files, console output, and other locations. Additionally, the product is occasionally referred to as the more generic "VoD Gateway" that describes its overall functionality. For all intents and purposes, please consider ABR2TS, VoD Gateway, and Media Transformer the same product.

Containerized Deployment

The CMT solution is deployed in a clustered environment utilizing the OpenShift Container Platform for node and container management. The solution consists of a set of microservices that run in Docker containers. These containers are deployed to the cluster nodes and managed via the Kubernetes

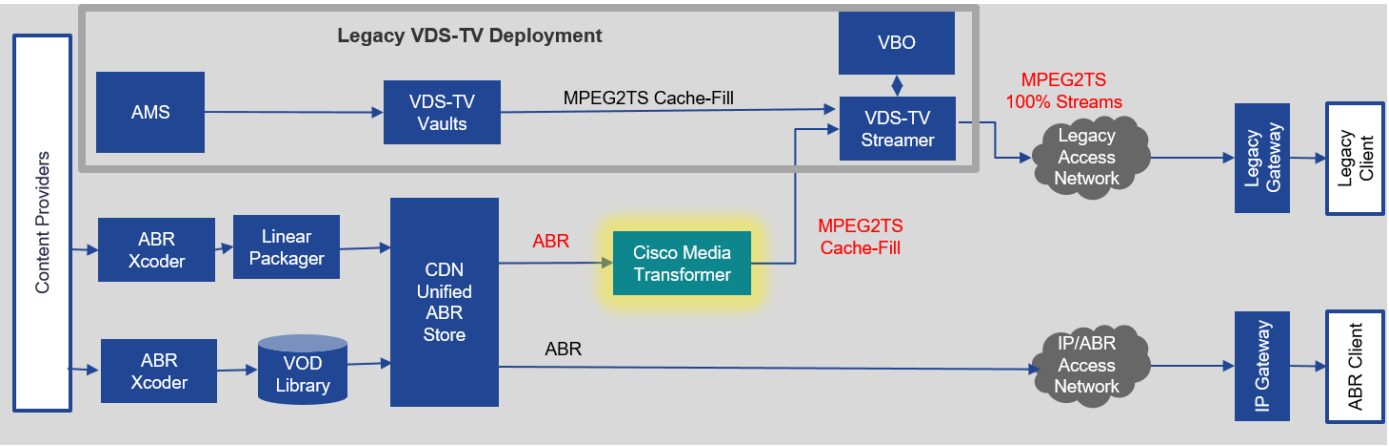
orchestration layer of the Openshift platform. This approach leverages the benefits and flexibility of container technology, such as high availability, auto-recovery, horizontal scalability, and ease of deployment.

Functional Overview

With respect to Cisco Media Transformer, the process starts in the following manner:

- The user of a set-top box requests specific content to the Video Back Office (VBO)
- VBO communicates with a master streamer, which selects the appropriate streamer that will serve the request. If the requested content is not cached on any of the streamers, then content will need to be pulled from the vaults, otherwise it will be served directly from the streamers.
- The system sees that the content is located at a URL and is not traditional VoD content. A conversion will need to take place.
- The system will pass the content URL, CBR bitrate, and starting/ending offsets to Media Transformer. The Media Transformer then fetches the manifest file from the CDN.
- The manifest provides a few key pieces of information to Media Transformer: representations, segment timeline, and segment location. Using the information in the manifest file along with the information provided in the request, Media Transformer is able to determine what segments need to be fetched from the CDN.
- The appropriate MPEG DASH segments are fetched, and transformed in real-time to be an MPEG-2 TS-compliant CBR stream, and delivered at a specified rate to the requesting system.
- The VDS-TV system will cache the CBR stream while delivering it to the QAM-based STB.

Figure 1-1 CMT Functional Overview



Worker Node Deployment

Worker nodes enable the core ABR to CBR conversion functionality within Media Transformer. ABR to CBR content transformation happens as part of the real-time streaming process. When the VDS-TV system detects that it does not have content in cache, it issues a request to Media Transformer to provide

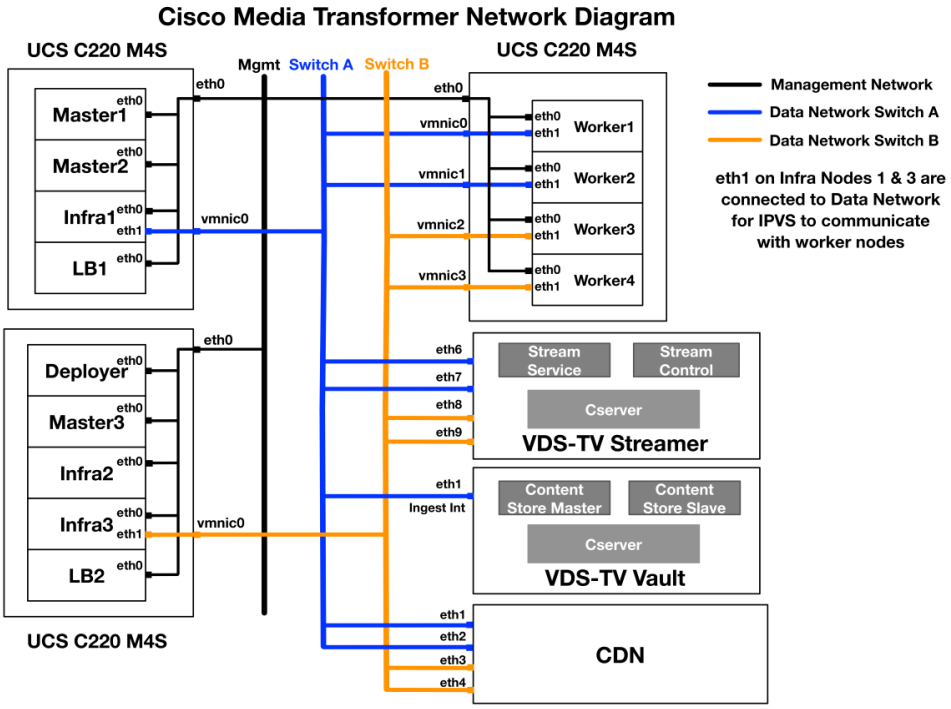
the content. This request will be directed to one of the Media Transformer pods for immediate processing. Since this is part of the real-time streaming process, the ABR content must be fetched, transformed, and delivered at a guaranteed rate specified by the VDS-TV system. A failure to deliver at rate will cause a VoD stream failure at the QAM or STB.

CMT Network Overview

Each UCS C220 M4 server is configured with four - 10GB network cards. The first two boards are connected to a Data A router, while the other two boards are connected to a Data B router. These data pathways are where the data from Media Transformer will be sent to the VDS-TV streamers. The purpose of having two data pathways is to provide high-availability functionality, so that if one router goes offline, then the other router will pick up the work and provide the required data stream.

Additionally, a 1GB network interface runs throughout the system to provide management functionality to Media Transformer - a task that requires less bandwidth than the data processing aspect. Figure 1-2 illustrates the Media Transformer network topology.

Figure 1-2 Media Transformer Network Diagram

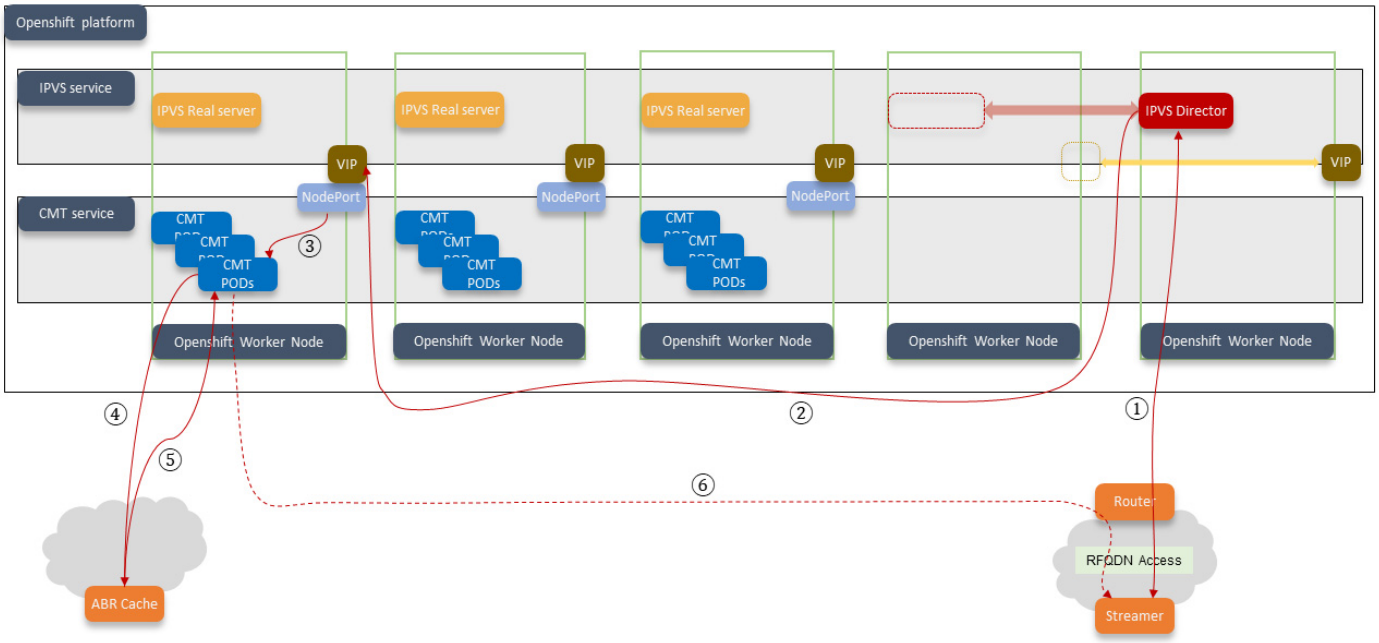


LB Request Example

All VDS-TV requests (API and client calls) to Media Transformer will first be sent to the IPVS load balancer. IPVS then redirects the calls onto different Media Transformer virtual machines (4 VMs exist per physical server). The Kubernetes instance on each virtual will then allocate the video processing load onto one of the five pod services (Docker containers) that it is managing.

After the Media Transformer pods have performed their work, they send the data back directly to the VDS-TV streamer, thereby bypassing the IPVS load balancer.

Figure 1-3 CMT Load Balance Solution



Virtual Machine Types

CMT consists of a set of virtual machines, each of which performs specific functions within the cluster and is packaged in an OVA file that encapsulates all functionality and optimal system configuration settings for each node type. An explanation of the virtual machine types and their resource requirements follows.

Master VMs

The OpenShift Master is the virtual machine that manages the entire cluster by communicating control messages to all of the cluster VM nodes. These services provide functionality related to pod management and the replication of nodes, authentication, data store, and scheduling. OpenShift Master services are packaged in the `CMT-Master-CSCOlxplat-CentOS-7.3.20170601-2.ova`. For details, see Installing the Virtual Machines, page 2-8.

Table 1-1 Master Node Virtual Machine Settings

Resource	Configuration
CPU	4 Cores
Memory	8GB
Disks	60GB disk space consisting of 2 - 30GB disks Operating System (30GB) & Docker (30GB)

Note We recommend 3 Master virtual machines within a cluster to fulfill high-availability requirements.

Deployer VMs

The OpenShift Deployer virtual machine stores the images and deployment scripts used to deploy and install all of the OpenShift images required for the initial cluster setup. This is a non-critical function for high-availability, so the cluster only needs a single Deployer node. OpenShift Deployer services are packaged in the `CMT-Deployer-201708181627-1.5.1.ova`. For details, see Installing the Virtual Machines, page 2-8.

Table 1-2 Deployer Node Virtual Machine Settings

Resource	Configuration
CPU	4 Cores
Memory	8GB
Disks	100GB disk space consisting of 2 - 50GB disks Operating System (50GB) & Docker (50GB)

Load Balancer VMs

The Load Balancer virtual machines define a node that is used to manage the OpenShift cluster. A load balancer Virtual IP is used to access the OpenShift cluster. OpenShift Load Balancer services are packaged in the `CMT-LB-CSCOlxplat-CentOS-7.3.20170601-2.ova`. For details, see Installing the Virtual Machines, page 2-8.

Table 1-3 Load Balancer Virtual Machine Settings

Resource	Configuration
CPU	2 Cores
Memory	4GB
Disks	20GB disk space Operating System (20GB)

Note We recommend 2 Load Balancer virtual machines within the cluster to fulfill high availability requirements. They will serve Master/Slave roles.

Infrastructure VMs

The Infrastructure virtual machines define a node that contains the IPVS load balancer (for CMT use), logging queue, and other infrastructure-related services such as those providing monitoring and alert functionality. The composition of these services will evolve over time. Infrastructure services are

packaged in the `CMT-Infra-CSCOlxplat-CentOS-7.3.20170601-2.ova`. For details, see Installing the Virtual Machines, page 2-8.

Table 1-4 Infra Virtual Machine Settings

Resource	Configuration
CPU	8 Cores
Memory	16GB
Disks	60GB disk space consisting of 2 - 30GB disks Operating System (30GB) & Docker (30GB)

Note A minimum of 3 infrastructure (Infra) nodes are required for a high availability system deployment.

Table 1-5 Recommended Infrastructure Service Allocation

Infrastructure VM 1	Infrastructure VM 2	Infrastructure VM 3
IPVS Director (Master)	Proxytoservice	IPVS Director (Standby)
Kafka	Kafka	Kafka
Zookeeper	Zookeeper	Zookeeper
Logstash	Logstash	Logstash

Worker VMs

OpenShift Worker virtual machines perform the primary functionality of CMT, which is to run multiple pods that convert adaptive bitrate (ABR) content to constant bitrate (CBR) content in real time with no latency or caching. As such, the CPU and memory resource requirements are considerable, relative to the rest of the system. OpenShift Worker services are packaged in `CMT-Worker-CSCOlxplat-CentOS-7.3.20170601-2.ova`. For details, see Installing the Virtual Machines, page 2-8.

Table 1-6 Worker Virtual Machine Settings

Resource	Configuration
CPU	7 Cores
Memory	60GB
Disks	60GB disk space consisting of 2 - 30GB disks Operating System (30GB) & Docker (30GB)

Note Swap memory will be set to 0 (meaning physical memory only is used) and hyper-threading should be disabled. Hyper-threading introduces some scheduling challenges into the system, so we have found that a more consistent throughput is achieved when using non-virtualized cores. Configuration instructions will be provided within this guide.

System Hardware Requirements

Media Transformer runs on general-purpose computing hardware, and is optimized for the Cisco Unified Computing System (UCS) server platform. Table 1-7 lists the recommended hardware configuration for a single Media Transformer server. For more detailed hardware requirements, refer to your Bill of Materials (BOM) or contact your Cisco Systems representative.

Note The recommended configuration for a CMT deployment is a minimum of 3UCS C220 M4 servers.

Table 1-7 Media Transformer Server Recommended Hardware Configuration

Description	Quantity
UCS C220 M4 Server	1
2.6GHz E5-2690 v4 CPUs	2
32GB DDR4 RAM	8
600 GB SAS 10K RPM HDD	2
Dual-port 10Gb Network Interface Cards	2

Terms and Definitions

Table 1-8 lists terms and definitions used in describing CMT or related concepts

Table 1-8 Terms and Definitions

Term	Definition
ABR2TS	The previous name for Cisco Media Transformer. This acronym still appears in various places throughout the installation process and therefore will also appear in this guide.
ABS	Adaptive Bitrate Streaming is where video content is streamed at the maximum rate and highest quality at which the network will allow at any given moment.
CBR	Constant Bitrate is where video content is streamed at a constant rate across a network.
Docker	A service used by Kubernetes to deploy containerized applications, such as the CMT solution.
IPVS	Linux IP Virtual Servers run on a host and act as a load balancer in front of a cluster of servers.
Kubernetes	Management system for containerized applications deployed across a cluster of nodes.

Table 1-8 Terms and Definitions

Term	Definition
Load Balancer Node	Two types of load balancers exist within the Media Transformer solution: 1) An IPVS Load Balancer directs external VDS-TV requests to different CMT virtual machines. 2) A Kubernetes instance on each virtual machine allocates the video processing load onto one of five Worker pods that it manages.
OMD Suite	Open Media Distribution Suite OMD is a suite of products for Service Providers to efficiently distribute and cache multi-screen video to managed & un-managed devices on managed & un-managed networks. Cisco Media Transformer is a part of OMD Suite.
POD	Are Docker containers that run microservices and that, in the Media Transformer solution, are managed by Kubernetes.
VDS-TV	The streamer component to which Media Transformer streams
Video BackOffice	Video BackOffice is a solution that provides a managed video control plane to service providers.

Installation Prerequisites

This chapter provides information about the prerequisites that must be met prior to installing CMT. It includes the following main topics:

- Configuring the UCS Servers, page 2-2
  - Configuring CIMC Access, page 2-2
  - Configuring Drives & Controllers, page 2-4
  - Configuring the CPU, page 2-6
  - Mapping Virtual Media, page 2-7
- Installing ESXi, page 2-7
  - Configuring ESXi, page 2-8
- Installing the Virtual Machines, page 2-8
  - Deploying OVA Images, page 2-8
  - Assigning IP Addresses, page 2-9
  - Configuring Swap Memory, page 2-11
  - Configuring NTP, page 2-12

Pre-installation Tasks

This section describes the tasks that should be performed prior to installing CMT.

Configuring the UCS Servers

A number of steps need to be performed to configure new UCS Servers for use in your CMT deployment. The following section will detail those procedures.

Configuring CIMC Access

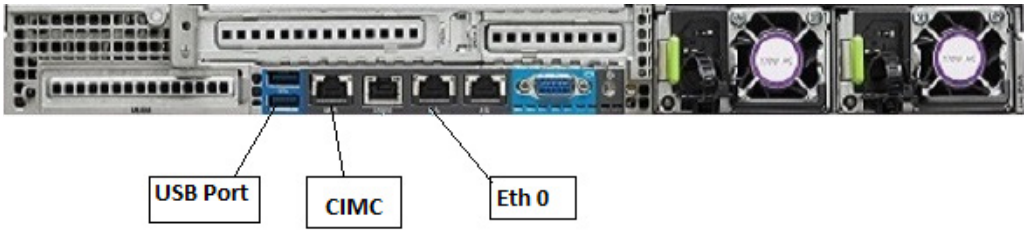
To initially configure your C220 M4 servers when you receive them:

Step 1 Verify that the server has all the expected hardware components as listed in the bill of materials (BOM).

Step 2 Attach an Ethernet cable to the management port of the UCS server. The CIMC (management port) is labeled on the server and is also show in the diagram below.

Note Two 10Gb/40Gb network interface cards (NICs) are normally inserted into the two PCI slots. They are not shown in the image below.

Figure 2-1 UCS Server Rear View

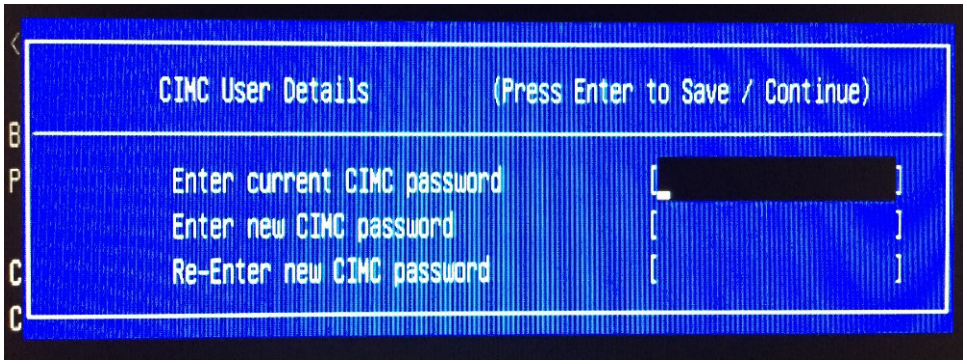


Step 3 Connect a VGA monitor, USB keyboard, and USB mouse to the server.

Step 4 Power up the server.

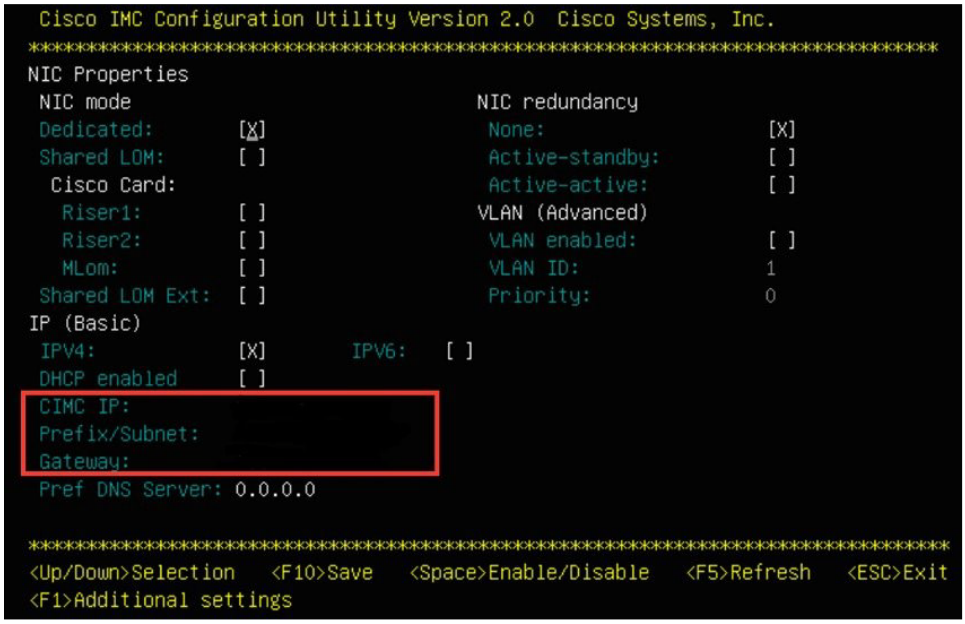
Step 5 Next, you will configure the Cisco Integrated Management Controller (CIMC). During the boot process, Press [F8]. This displays a dialog where you will be able to change the CIMC password for gaining access to the UCS server. The default initial password is “password”. Type the new password and press [Enter] to save it into the system.

Figure 2-2 CIMC Set Password Page



Step 6 To assign the CIMC IP address by which you will manage the server, enter your IP address, subnet mask, and gateway addresses. Those should be provided to you by your network administrators.

Figure 2-3 CIMC Configuration Utility Page



Step 7 Set NIC redundancy to “None [X]”.

Step 8 Press [F10] to save the settings.

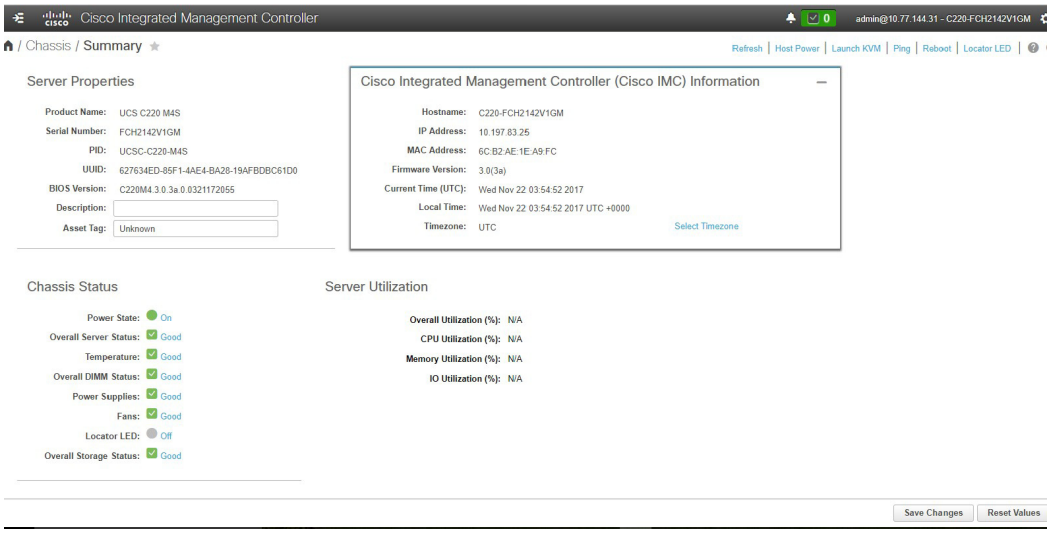
Step 9 Confirm that the IP address associated with the CIMC port has been properly set by pinging it.

Step 10 Test that the CIMC interface can be reached by your Web browser using the following address:

`https://{CIMC_IP}`

Step 11 The credentials will be username: `admin` and the password will be the password you configured earlier or `password` if the default value has not been changed.

Figure 2-4 CIMC Chassis Summary View

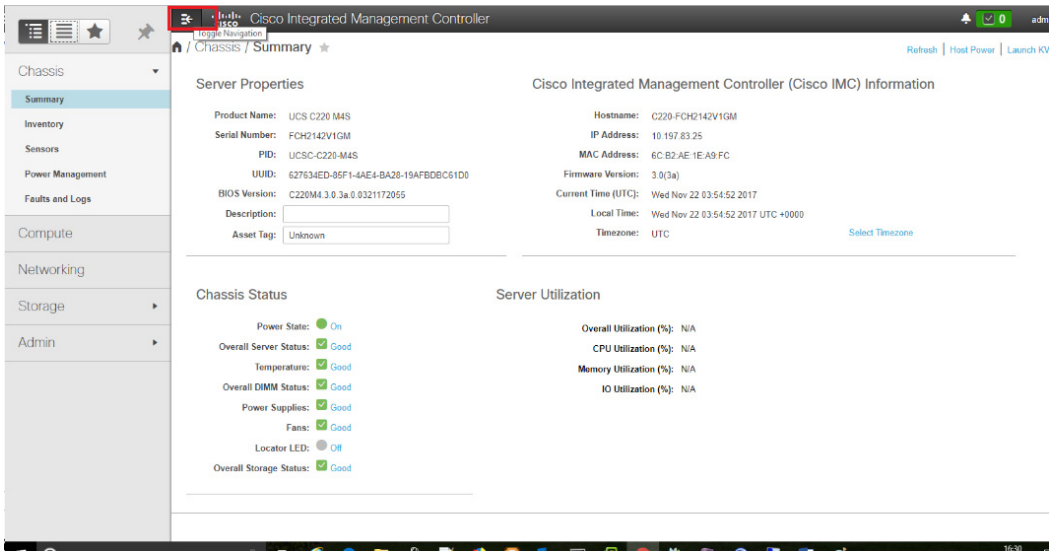


Configuring Drives & Controllers

Now that you have access to the Web-based CIMC user interface, you can configure the UCS server drives and controllers.

Step 1 Within the CIMC, click the **Navigation Toggle** at the upper-left corner of the user interface. The toggle is just left of the Cisco logo.

Figure 2-5 CIMC Chassis Summary View - Toggle

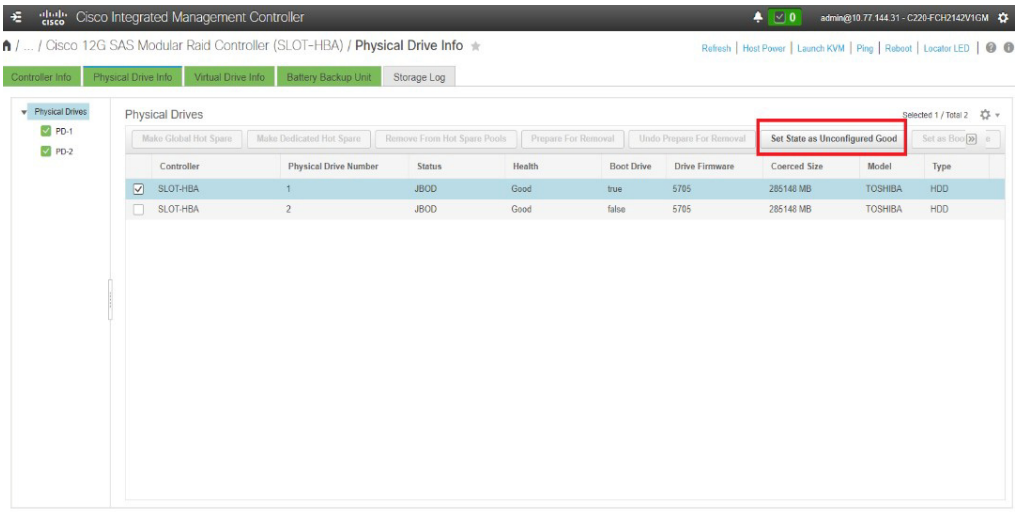


Step 2 Navigate to **Storage > Cisco 12G SAS Modular Raid Controller > Physical drive info**

Step 3 Select the drives to be configured.

Step 4 Click **Set State as Unconfigured Good**.

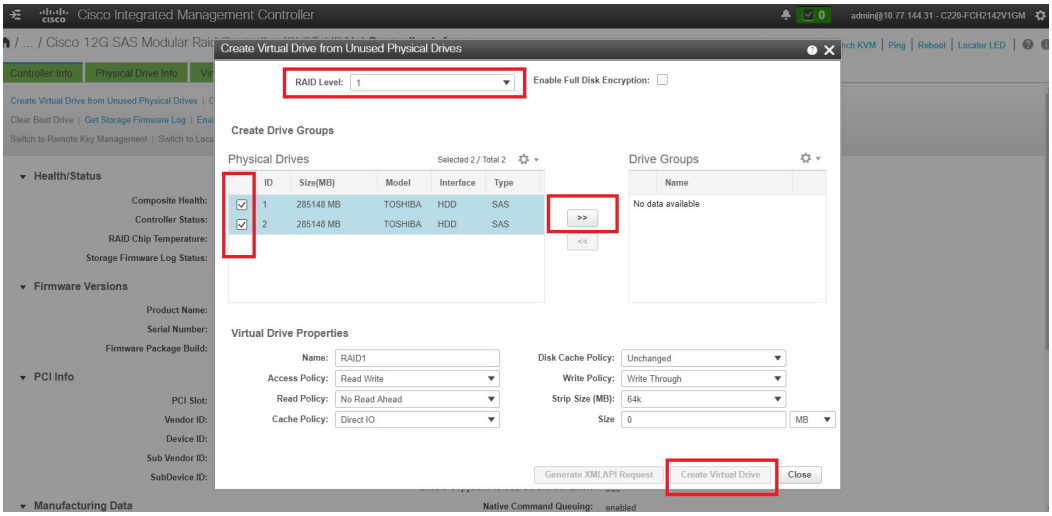
Figure 2-6 CIMC Physical Drive Info



Step 5 The status column should now show “Unconfigured Good” instead of “JBOD”.

Step 6 Navigate to **Controller Info > Create Virtual Drive from unused Physical Drives**.

Figure 2-7 CIMC Create Virtual Drive Dialog



Step 7 Set the RAID Level as 1. This RAID level mirrors drives to provide data redundancy.

Step 8 Select both Physical Drives and move them over to the Drive Groups table by clicking **>>**.

Step 9 Confirm that the size value is correct.

Step 10 Click **Create Virtual Drive**.

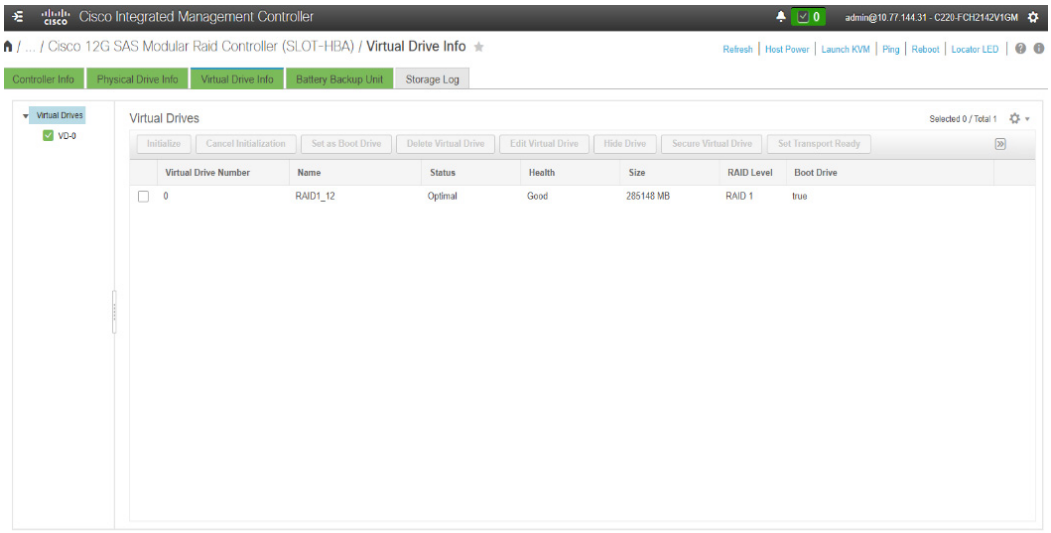
Step 11 Navigate to the **Virtual Drive Info** tab.

Step 12 Select the new Raid 1 virtual drive.

Step 13 Click **Initialize**. Choose the **Fast Initialize** option.

Step 14 Click **Set as Boot Drive**.

Figure 2-8 CIMC Virtual Drive Configuration is Complete



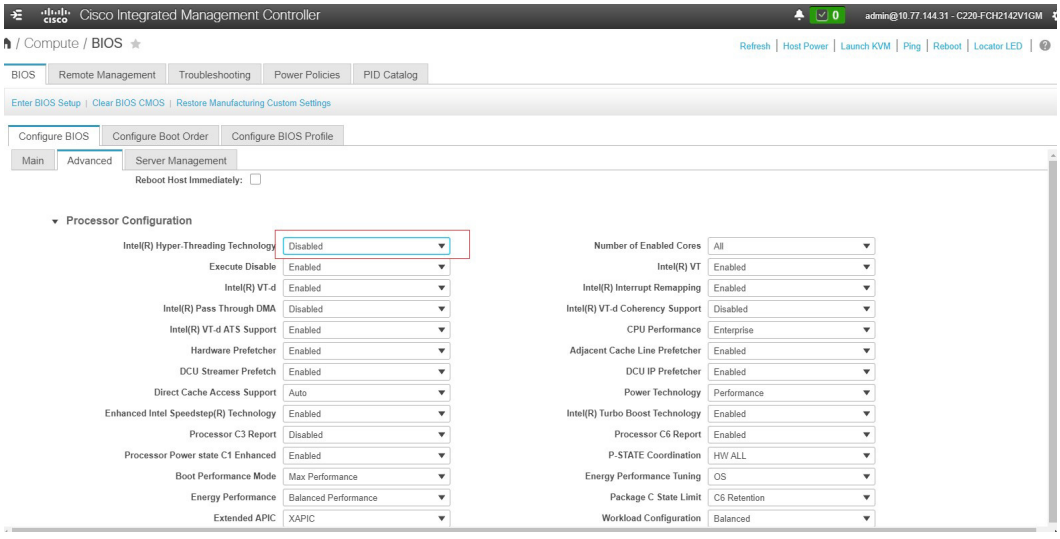
Configuring the CPU

Next, the CPU should be configured to turn off Hyper-threading. This setting provides a more consistent throughput to the cluster.

Step 1 Click the **Navigation Toggle** near the upper-left corner of the user interface.

Step 2 Navigate to **Compute > Configure BIOS > Advanced**.

Figure 2-9 CIMC - Configure Bios - Advanced



Step 3 In the Processor Configuration section, set Intel(R) Hyper-Threading Technology to **Disabled**.

Step 4 Navigate to the **Configure Boot Order** tab.

Step 5 Click the **Advanced** tab.

Step 6 Click **Add Virtual Media**.

Step 7 Click **Save Changes**.

Mapping Virtual Media

The following section will explain the procedures for mapping a storage drive connect to a laptop or PC as a virtual media device. The storage device you choose should be a removable drive with a bootable ESXi image on it.

Step 1 In the CIMC interface, click **Launch KVM > HTML based KVM**.

Step 2 Click the displayed URL to launch the console.

Step 3 Power off the device by clicking **Power > Power Off System** Wait for the next user interface to appear.

Step 4 Click **Virtual Media : Activate Virtual Devices**.

Step 5 Click **Virtual Media > Map CD/DVD** (or **Map Removable Disk** if that is more appropriate).

Step 6 Click **Map Drive**. Next, you will need install ESXi.

Installing ESXi

ESXi is a Hypervisor made by VMWare that allows you to run virtual machines directly on bare metal. The following section briefly explains the process for installing this software. To install ESXi:

Step 1 Click **Power > Power ON System**.

Step 2 Once the device has booted, an auto-installation process will begin and command-line output will be displayed on the console.

Step 3 Press **[Enter]** at the Welcome to the VMWare ESXi 6.0.0 Installation dialog.

Step 4 Review and then press **[F11]** to accept the End User Licensing Agreement.

Step 5 Press **[Enter]** to select a disk to install ESXi onto.

Step 6 Press **[Enter]** to accept a US Default keyboard layout.

Step 7 Set your root password, confirm the password, and press **[Enter]** to save it.

Step 8 Press **[F11]** to commence the ESXi installation.

Warning **The process of installing ESXi will repartition the selected drive.**

Step 9 Once installation has completed, you should receive a confirmation message. Press **[Enter]** to reboot and to start ESXi.

Step 10 The server will boot to the ESXI user interface.

Configuring ESXi

After ESXI has booted up, you will need to configure your server.

Step 1 Press **[F2]**, type the root password, and press **[Enter]**.

Step 2 Navigate to **Configure Management Network** and press **[Enter]**.

Step 3 Navigate to **IPv4 Configuration** and press **[Enter]**.

Step 4 On the IPv4 Configuration dialog, select the **Set static IPv4 address and network configuration** option. Press **[Space]**.

Step 5 Set the IPv4 address, Subnet Mask, and Default Gateway. Those details should have been provided by your network administrator.

Step 6 Press **[Enter]** to confirm the values.

Step 7 Press **[Esc]** to exit the **Configure Network Management** dialog.

Step 8 Press **[Y]** to confirm the changes and to restart the management network.

Installing the Virtual Machines

The following instructions were created using UCS servers running ESXi 6.0. Each virtual machine type requires its own OVA image. When deployed, that image will create the respective virtual machines and configure all RAM, CPU, and Disk requirements for that node. The OVA file for each node type is listed below:

- Deployer Node requires `CMT-Deployer-201708181627-1.5.1.ova`
- Master Node requires `CMT-Master-CSCOlxplat-CentOS-7.3.20170601-2.ova`
- Worker Node requires `CMT-Worker-CSCOlxplat-CentOS-7.3.20170601-2.ova`
- Infrastructure Node requires `CMT-Infra-CSCOlxplat-CentOS-7.3.20170601-2.ova`
- Load Balancer Node requires `CMT-LB-CSCOlxplat-CentOS-7.3.20170601-2.ova`

Deploying OVA Images

The process for deploying OVA images is identical regardless of the image being deployed.

Step 1 In vSphere, navigate to **File > Deploy OVF Template**.

Step 2 Browse and select the appropriate **.ova* file for the node type that you want to deploy.

Step 3 Click **Next**.

Step 4 Examine the License Agreement. Click **Accept** and then **Next**.

Step 5 For Name and Location, specify a name and location for the deployed template.
One suggested naming convention follows, where N=number to identify multiple nodes:

- CMT-Deployer
- CMT-MasterN
- CMT-InfraN
- CMT-LoadBalancerN
- CMT-WorkerN

Note After deploying Worker OVA files, you must change the virtual machine swap memory to 0 for those nodes.

Step 6 Click **Next**.

Step 7 For Disk Format, select the **Thin Provision** option and then click **Next**.

Step 8 For Network Mapping, select the appropriate network connection. Click **Next**.

Step 9 For Ready to Complete, review the deployment settings to ensure that they are correct. Click **Finish**.

Step 10 Repeat all of these steps for each node in the cluster.

Assigning IP Addresses

Next, you will need to assign IP addresses to each node in the cluster via the NetworkManager Text User Interface (nmtui) tool. The process will be identical in each instance, regardless of node type.

Step 1 Power on the VM you wish to configure.

Step 2 Open the console.

Step 3 Log in as root.

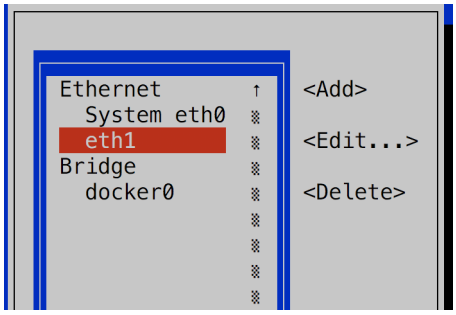
Step 4 Run the `nmtui` command.

Step 5 Select **Edit a connection**. Press **[Enter]**.

Figure 2-10 NetworkManager TUI - Edit a connection

Step 6 Select `System eth0`. Click **[Edit]**. Press **[Enter]**.

Figure 2-11 NetworkManager TUI - Selecting eth0



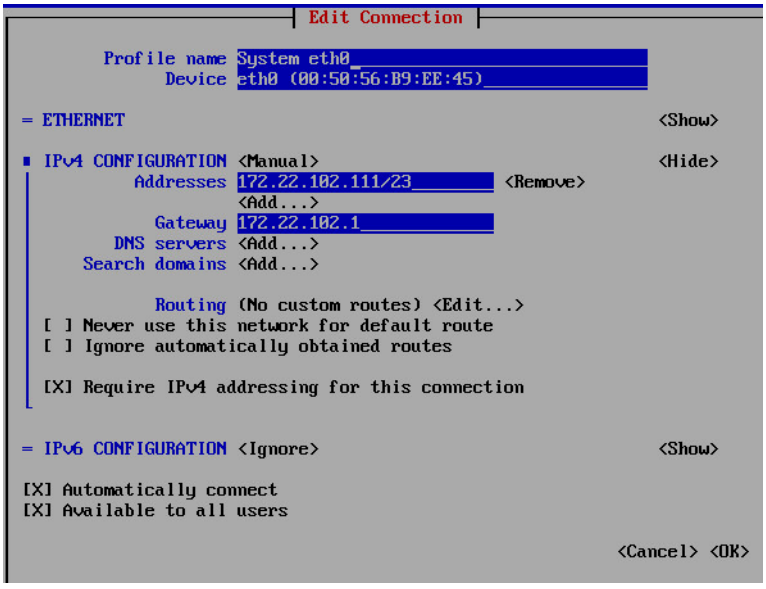
Note Worker and Infra nodes (only) will also need to be configured using the Eth1 (10/40Gb) interface. Eth0 is strictly for cluster management.

Step 7 On the Edit Connection page, update the `Adddresses` field with the IP address and subnet. For example: `172.22.102.111/23`

Step 8 Update the `Gateway IP` value.

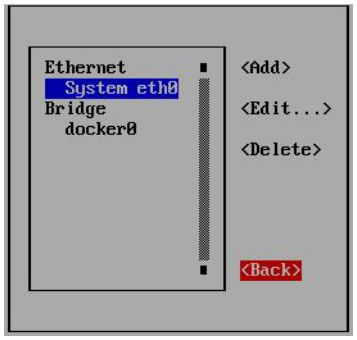
Note When you are configuring the Eth1 interface (only), you must enable the: **[X] Never use this network for default route** option.

Figure 2-12 NetworkManager TUI - Editing Connection Details



Step 9 Press **[OK]**. Press **[Back]**. Press **[Enter]**.

Figure 2-13 NetworkManager TUI - Stepping Back in Menus



Step 10 Select `Quit` and press **[Enter]**.

Figure 2-14 Network Manager TUI - Quitting

Step 11 On the console, run the `service network restart` command.

Step 12 Ping the IP address of the node to make sure that it is reachable.

Step 13 Repeat this process for each virtual machine.

Configuring Swap Memory

The worker node virtual machines must have their swap memory set to 0. This action is performed in the following manner:

Step 1 Within vSphere, navigate to the Virtual Machine Properties.

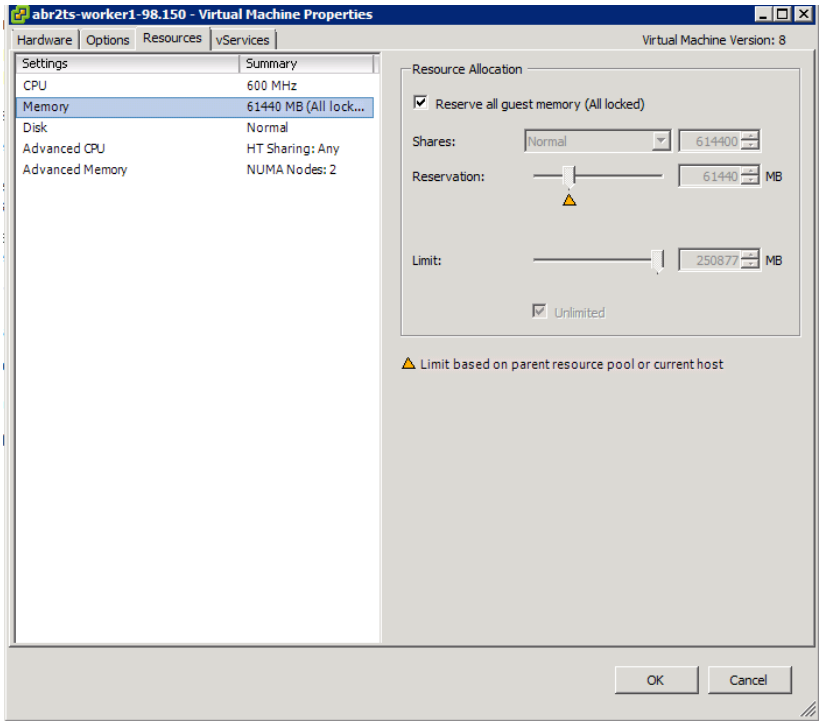
Step 2 Click the **Resources** tab.

Step 3 Click the **Memory** setting,

Step 4 Set the Reservation value to 61440MB.

Step 5 Enable the **Reserve all guest memory (All locked)** option.

Figure 2-15 Virtual Machine Properties - Memory



Step 6 Click **OK**.

Configuring NTP

You must initially configure the NTP service on the deployer node to correctly synchronize the date and time for the system. The process for configuring NTP services on the deployer node is as follows:

Step 1 SSH into the deployer node as the root user.

Step 2 Edit **/etc/ntp.conf** to add the following line:

```
server {ntp server}
```

Step 3 Restart the NTP service by using the command:

```
service ntpd restart
```

Step 4 Synchronize the time and date from the NTP server by using the command:

```
ntpdate -u {ntp server}
```

Installation

This chapter provides instructions for installing the Media Transformer (CMT) Video on Demand gateway. It includes the following main topics:

- Editing the Inventory File, page 3-1
- Performing the Installation, page 3-9
- Load Images into Docker Registry, page 3-25
- Create the ABR2TS Project Namespace, page 3-34
- Logging Queue Deployment, page 3-36
- Starting VoD Gateway & Fluentd, page 3-42
- Configuring IPVS, page 3-45
- Installing the Monitoring Stack, page 3-54

Note We recommend that you carefully review the Pre-installation Tasks in Chapter 2, "Installation Prerequisites" prior to beginning the installation process.

Editing the Inventory File

The inventory file contains many of the server settings that establish key aspects of your CMT installation. Changes must be made to the inventory file prior to using it to install the OpenShift cluster.

Step 1 Within the command line, SSH as root into the deployer node.

Step 2 Change directory:

```
cd ivp-coe/
```

Step 3 Edit the inventory file:

```
vi abr2ts-inventory
```

Step 4 Change the values shown in bold in the inventory file below. In some instances, important comments are also in bold to highlight them.

```
#####################################################################################################################
# SAMPLE inventory (owner: SAMPLE (sample) (sample@cisco.com))
#
```

```
# For a full set of options and values see:
# ../openshift-ansible/inventory/byo/hosts.origin.example
#
# When adding labels to your nodes please follow the guidance in this document:
# https://wiki.cisco.com/pages/viewpage.action?pageId=64776674
#############################################################################################################


[all:vars]
cluster_timezone=UTC

openshift_master_default_subdomain=cmtlab-dns.com

### In a HA LB cluster     | the <MASTER-API-HOSTNAME> will the NAME of the keepalived_vip (eg 'cmt-vip')
### In a non HA LB cluster | the <MASTER-API-HOSTNAME> will the NAME of the main LB (eg 'cmt-lb1')
### With a single Master   | the <MASTER-API-HOSTNAME> will the NAME of the MASTER node (eg 'cmt-master11')
### In all cases use NAMEs NOT IP Addresses.
openshift_master_cluster_hostname=cmt-osp-cluster.cmtlab-dns.com
openshift_master_cluster_public_hostname=cmt-osp-cluster.cmtlab-dns.com

### In a HA LB cluster     | set the 10.84.73.101 to the VIP ip and configure the <MASTER-API-FQDN> above
###                        | COE is 'magic' and will work out everything for you from these settings
### Access the UI          | insert the VIP and <MASTER-API-FQDN> in the hosts file cwon you local devicei
###                        | or define it in your DNS (no not the Openshift clusters DNS or host file)
### keepalived_vrrpid      | is an integer between 1-255 for the VRRPID that is unique to the clusters subnet
keepalived_vip=172.22.102.244                          #Load balancer VIP
#keepalived_interface=<node interface on load balancers for VIP>
keepalived_interface=eth0
keepalived_vrrpid=172

yumrepo_url=http://172.22.102.170/centos/7/        # Deployer IP

#values may be a comma seperated list to specify additional registries
#the deployers registry (eg <DEPLOYER-IP>:5000) must be the first entry in this list
openshift_docker_additional_registries=172.22.102.170:5000
openshift_docker_insecure_registries=172.22.102.170:5000
openshift_docker_blocked_registries=docker.io

#optional list of ntp server(s) accessable to all cluster nodes (with alternative examples)
#setting a value overrides the default upstream ntp server list
##ntp_servers=["<NTP_SERVER1>","<NTP_SERVER2>","<NTP_SERVER3>"]
ntp_servers=["172.22.116.17"]

openshift_disable_check=disk_availability,docker_storage

# To use different subnet for openshift install other than 172.30.0.0/16 uncomment the below line and update
subnet.
#openshift_portal_net=172.20.0.0/16

# Create an OSEv3 group that contains the masters and nodes groups
[OSEv3:children]
masters
nodes
etcd
lb
gluster
new_nodes


# Set variables common for all OSEv3 hosts
[OSEv3:vars]
ansible_ssh_user=root
ansible_become=true
debug_level=2
```

```
deployment_type=origin
openshift_image_tag=v1.5.1
openshift_install_examples=false
openshift_master_cluster_method=native
openshift_pkg_version=-1.5.1
openshift_release=v1.5
logrotate_scripts=[{"name": "syslog", "path":
"/var/log/cron\n/var/log/maillog\n/var/log/messages\n/var/log/secure\n/var/log/spooler\n/var/lib/docker/conta
iners/*/*-json.log\n", "options": ["daily", "rotate 10", "size 100M", "compress", "sharedscripts",
"missingok"], "scripts": {"postrotate": "/bin/kill -HUP `cat /var/run/syslogd.pid 2> /dev/null` 2> /dev/null
|| true"}}]


# Password Identity Provider
# To enable it un-comment the 2 variables in place
# Generated Password is stored in ivp-coe/.originrc_<MASTER-API-FQDN>
# If re-installation/upgrade is run then the old file is backed up and a new password file is generated
# ### openshift_master_identity_providers=[{'name': 'htpasswd_auth', 'login': 'true', 'challenge': 'true',
'kind': 'HTPasswdPasswordIdentityProvider', 'filename': '/etc/origin/master/htpasswd'}]
# ### openshift_master_htpasswd_file="/tmp/.authT1"


##############################################################################################################
###

### Stardard OpenShift Router configuration FOR POD/Container ipfailover/router ###
### To use the standard Openshift Origin Router on port 80/443
### set the value below to true and ensure the following IPFailOver section is commented out

openshift_hosted_manage_router=false
openshift_hosted_router_selector='region=infra'

##############################################################################################################
###

### Bespoke IPFailOver configuration FOR POD/Container ipfailover/router ###
### To use something other than the standard Openshift Origin Router on port 80/443
### set the value above to false and ensure the following IPFailOver section is uncommented

### In this section you assign router(s) with ipfailover, creating VIP(s) to front the router
### You can setup one or more ipfailover/router pod combinations. Multiple routers can not run on the same
node.

### list_item?=     | an arbitrary label used to construct the final ipfs= data structure. One per line
### label           | a simple identifier for each ipfs ruleset. It is also the =V value in the selector K=V
pair
###                 | do not change a label after it has been used (unless you manually delete the associated
configuration)
### K=V Selector    | the node selector identifing the minions the ipfailover/router pod can run on.
### replicas        | the number of ipfailover/router pod(s) to run for this ruleset. (ideally 1 less than
node set)
### VIP(s)          | Virtual IPs to fron this ruleset
### port            | the port to expose and listen on for this ruleset (this could be 8080 instead of the
usual 80)
### vrrp_id_offset | a unique identify on the local subnet to prevent collisions between rulesets (value: 1 -
255)
### NIC             | the network interface card to listen on. (usually eth0)

### delete_ipfs_config_before_create=false
### list_item1=[ "label", "K=V selector", replicas, "VIP(s)", port, vrrp_id_offset, "NIC" ]
### list_item2=[ "label", "K=V selector", replicas, "VIP(s)", port, vrrp_id_offset, "NIC" ]
### ...
### ipfs=[ <list_item1> , <list_item2> , ... ]
```

```
#############################################################################################################################
###


#Openshift Registry Options
openshift_hosted_manage_registry=false

#Openshift Metrics deployment (https://hawkular-metrics.<DOMAIN>/hawkular/metrics)
openshift_hosted_metrics_deploy=false
# Note that <DOMAIN> must have the same value set for 'openshift_master_default_subdomain'
#openshift_hosted_metrics_public_url=https://hawkular-metrics.cmtlab-dns.com/hawkular/metrics
#openshift_hosted_metrics_deployer_prefix=172.22.102.170:5000/openshift/origin-
#openshift_hosted_metrics_deployer_version=v1.5.1

#Openshift Logging deployment (https://kibana.<DOMAIN>)
#openshift_hosted_logging_deploy=false
#openshift_hosted_logging_deployer_prefix=172.22.102.170:5000/openshift/origin-
#openshift_hosted_logging_deployer_version=v1.5.1

#Openshift-ansible docker options get setup here:
#Modify according to your needs
#Defaults:
#    log-driver:journald
#    dm.basesize: 10G
openshift_docker_options="--log-driver=json-file --log-opt=max-size=200m"
selinux_fix_textreloc=false


# Enable origin repos that point at Centos PAAS SIG, defaults to true, only used by deployment_type=origin
# This should be false for the deployer
openshift_enable_origin_repo=false

# Origin copr repo; Setup Only if different from the yumrepo_url
#openshift_additional_repos=[{'id': 'openshift-origin-copr', 'name': 'OpenShift Origin COPR', 'baseurl':
'<YUMREPO_PAAS_URL>', 'enabled': 1, 'gpgcheck': 0}]


#host group for masters
[masters]
cmt-master1 ansible_ssh_host=172.22.102.143 openshift_ip=172.22.102.143 openshift_public_ip=172.22.102.143
openshift_public_hostname=cmt-master1 openshift_hostname=cmt-master1 openshift_schedulable=false
cmt-master2 ansible_ssh_host=172.22.102.164 openshift_ip=172.22.102.164 openshift_public_ip=172.22.102.164
openshift_public_hostname=cmt-master2 openshift_hostname=cmt-master2 openshift_schedulable=false
cmt-master3 ansible_ssh_host=172.22.102.169 openshift_ip=172.22.102.169 openshift_public_ip=172.22.102.169
openshift_public_hostname=cmt-master3 openshift_hostname=cmt-master3 openshift_schedulable=false


#cmt-master ansible_ssh_host=<MASTER3-IP> openshift_ip=<MASTER3-IP> openshift_public_ip=<MASTER3-IP>
openshift_public_hostname=cmt-master openshift_hostname=cmt-master openshift_schedulable=false

[masters:vars]
reboot_timeout=300


#host group for minions
[minions]
cmt-worker1 ansible_ssh_host=172.22.102.152 openshift_ip=172.22.102.152 openshift_public_ip=172.22.102.152
openshift_public_hostname=cmt-worker1 openshift_hostname=cmt-worker1 openshift_node_labels="{'region':
'infra', 'cisco.com/type':'backend', 'network.cisco.com/eth0':'172.22.102.152',
'network.cisco.com/eth1':'192.169.131.2', 'network.cisco.com/lo':'127.0.0.1'}" openshift_schedulable=true
cmt-worker2 ansible_ssh_host=172.22.102.153 openshift_ip=172.22.102.153 openshift_public_ip=172.22.102.153
openshift_public_hostname=cmt-worker2 openshift_hostname=cmt-worker2 openshift_node_labels="{'region':
'infra', 'cisco.com/type':'backend', 'network.cisco.com/eth0':'172.22.102.153',
'network.cisco.com/eth1':'192.169.131.3', 'network.cisco.com/lo':'127.0.0.1'}" openshift_schedulable=true
```

```
cmt-worker3 ansible_ssh_host=172.22.102.250 openshift_ip=172.22.102.250 openshift_public_ip=172.22.102.250
openshift_public_hostname=cmt-worker3 openshift_hostname=cmt-worker3 openshift_node_labels="{'region':
'infra', 'cisco.com/type':'backend', 'network.cisco.com/eth0':'172.22.102.250',
'network.cisco.com/eth1':'192.169.131.4', 'network.cisco.com/lo':'127.0.0.1'}" openshift_schedulable=true
#cmt-worker4 ansible_ssh_host=172.22.98.117 openshift_ip=172.22.98.117 openshift_public_ip=172.22.98.117
openshift_public_hostname=cmt-worker4 openshift_hostname=cmt-worker4 openshift_node_labels="{'region':
'infra','cisco.com/type':'backend', 'network.cisco.com/eth0':'172.22.98.117',
'network.cisco.com/eth1':'192.169.150.8', 'network.cisco.com/lo':'127.0.0.1'}" openshift_schedulable=true

cmt-infra1 ansible_ssh_host=172.22.102.58 openshift_ip=172.22.102.58 openshift_public_ip=172.22.102.58
openshift_public_hostname=cmt-infra1 openshift_hostname=cmt-infra1 openshift_node_labels="{'region': 'infra',
'infra.cisco.com/type':'infra', 'cisco.com/type':'master', 'network.cisco.com/eth0':'172.22.102.58',
'network.cisco.com/eth1':'192.169.131.5', 'network.cisco.com/lo':'127.0.0.1'}" openshift_schedulable=true
cmt-infra2 ansible_ssh_host=172.22.102.61 openshift_ip=172.22.102.61 openshift_public_ip=172.22.102.61
openshift_public_hostname=cmt-infra2 openshift_hostname=cmt-infra2 openshift_node_labels="{'region': 'infra',
'infra.cisco.com/type':'infra', 'cisco.com/type':'infra', 'network.cisco.com/eth0':'172.22.102.61',
'network.cisco.com/eth1':'192.169.131.6', 'network.cisco.com/lo':'127.0.0.1'}" openshift_schedulable=true
cmt-infra3 ansible_ssh_host=172.22.102.65 openshift_ip=172.22.102.65 openshift_public_ip=172.22.102.65
openshift_public_hostname=cmt-infra3 openshift_hostname=cmt-infra3 openshift_node_labels="{'region': 'infra',
'infra.cisco.com/type':'infra', 'cisco.com/type':'master', 'network.cisco.com/eth0':'172.22.102.65',
'network.cisco.com/eth1':'192.169.131.7', 'network.cisco.com/lo':'127.0.0.1'}" openshift_schedulable=true

#cmt-node3 ansible_ssh_host=<NODE3-IP> openshift_ip=<NODE3-IP> openshift_public_ip=<NODE3-IP>
openshift_public_hostname=cmt-node3 openshift_hostname=cmt-node3 openshift_node_labels="{'region': 'infra'}"
openshift_schedulable=true

[minions:vars]
reboot_timeout=300


## The [gluster] group is used when configuring glusterfs storage
## Ansible will expect this block to exist in the inventory EVEN if it is empty.
## DO NOT comment or remove [gluster], but it can be an empty group
[gluster]
#cmt-node4 ansible_ssh_host=<NODE4-IP> openshift_ip=<NODE4-IP> openshift_public_ip=<NODE4-IP>
openshift_public_hostname=cmt-node4 openshift_hostname=cmt-node4 openshift_node_labels="{'app': 'gluster'}"
openshift_schedulable=false
#cmt-node5 ansible_ssh_host=<NODE5-IP> openshift_ip=<NODE5-IP> openshift_public_ip=<NODE5-IP>
openshift_public_hostname=cmt-node5 openshift_hostname=cmt-node5 openshift_node_labels="{'app': 'gluster'}"
openshift_schedulable=false

## The [gluster:vars] group is used when configuring glusterfs storage
## Ansible will expect this block to exist in the inventory EVEN if it is empty.
## DO NOT comment or remove [gluster:vars], but it can be an empty group
[gluster:vars]
## gluster physical disk device and partition number
#gluster_pv_device=sdc  ### describe an empty disk device  ### SET THIS and remove this comment
#gluster_pv_part=1      ### and partion for gluster store  ### SET THIS and remove this comment
## gluster brick [ <brick numeric id>, "<size>" ]
#gluster_brick01=[ 1, "5G" ]
#gluster_bricks=[gluster_brick01,gluster_brick02]
## gluster volume [ <volume numeric id>, "<volume name>", <brick numeric id>, <replicas> ]
#gluster_volume01=[ 1, "test01", 1, 3 ]
#gluster_volumes=[gluster_volume01,gluster_volume02,gluster_volume03,gluster_volume04]
#reboot_timeout=<TIMEOUT IN SECONDS>


#host group for nodes
[nodes:children]
masters
minions
gluster

[nodes:vars]
```

```
pv_device=sdb
pv_part=1


## The [new_masters] group is used when performing a scaleup process.
## Ansible will expect this block to exist in the inventory EVEN if it is empty.
## DO NOT comment or remove [new_masters], but it can be an empty group
[new_masters]


## The [new_masters:vars] group is used when performing a scaleup process.
## Ansible will expect this block to exist in the inventory EVEN if it is empty.
## DO NOT comment or remove [new_masters:vars], but it can be an empty group
[new_masters:vars]


## The [new_minions] group is used when performing a scaleup process.
## Ansible will expect this block to exist in the inventory EVEN if it is empty.
## DO NOT comment or remove [new_minions], but it can be an empty group
[new_minions]

#cmt-node5 ansible_ssh_host=10.197.86.240 openshift_ip=10.197.86.240 openshift_public_ip=10.197.86.240
openshift_public_hostname=cmt-node5 openshift_hostname=cmt-node5 openshift_node_labels="{'region':
'infra','cisco.com/type':'backend'}" openshift_schedulable=true


## The [new_minions:vars] group is used when performing a scaleup process.
## Ansible will expect this block to exist in the inventory EVEN if it is empty.
## DO NOT comment or remove [new_minions:vars], but it can be an empty group
[new_minions:vars]


## The [new_nodes:children] group is used when performing a scaleup process.
## Ansible will expect this block to exist in the inventory ALWAYS.
## DO NOT comment or remove [new_nodes:children], new_master or new_minions
[new_nodes:children]
new_masters
new_minions


## The [new_nodes:vars] group is used when performing a scaleup process.
## Ansible will expect this block to exist in the inventory EVEN if it is empty.
## DO NOT comment or remove [new_nodes:vars], but it can be an empty group
[new_nodes:vars]
pv_device=sdb
pv_part=1


#host group for nfs servers
#[nfs]
#<IPADDR> ansible_ssh_host=<IPADDR> openshift_ip=<IPADDR> openshift_public_ip=<IPADDR>
openshift_public_hostname=<IPADDR> openshift_hostname=<IPADDR>

#[nfs:vars]
#number_of_pvs=<NUM>


[etcd:children]
masters
new_masters

[etcd:vars]
```

```
[lb]
cmt-lb1 ansible_ssh_host=172.22.102.241 openshift_ip=172.22.102.241 openshift_public_ip=172.22.102.241
openshift_public_hostname=cmt-lb1 openshift_hostname=cmt-lb1 ha_status=MASTER
cmt-lb2 ansible_ssh_host=172.22.102.243 openshift_ip=172.22.102.243 openshift_public_ip=172.22.102.243
openshift_public_hostname=cmt-lb2 openshift_hostname=cmt-lb2 ha_status=SLAVE
#cmt-lb2 ansible_ssh_host=<LB2-IP> openshift_ip=<LB2-IP> openshift_public_ip=<LB2-IP>
openshift_public_hostname=cmt-lb2 openshift_hostname=cmt-lb2 ha_status=SLAVE
[lb:vars]


[deployer]
cmt-deployer ansible_ssh_host=172.22.102.170 openshift_ip=172.22.102.170 openshift_hostname=cmt-deployer

[deployer:vars]

[root@platform ~]#
```

Step 5 Save the file.

Increase Timeout for Docker Image Load

In order to perform a successful installation, you must increase the timeout values for Docker Image Load and Reboot.

Step 1 Within the Linux command line, navigate to the `ivp-coe` directory.

Step 2 Edit the `load_registry.yml` file to increase the timeout value as shown in bold below.

```
root@platform ivp-coe]# vi load_registry.yml

tasks:
   - when: openshift_docker_additional_registries is defined
     block:
     - name: Docker Load | load image from /ivp-coe/registry archives
       docker_image:
         name:
"{{openshift_docker_insecure_registries.split(',')[0]}}/{{item.prefix}}{{item.name}}"
         tag: "{{item.tag}}"
         load_path: "{{item.file}}"
         timeout: 300
```

Step 3 Save the file.

Update the dnsmasq

When you run the Ansible playbook `run_installation` in a coming step, the dnsmasq will be updated on all of the cluster nodes to provide DNS forwarding capabilities to the cluster. Update the server domain name and IP address shown in bold to allow the update to proceed correctly.

dnsmasq file to update:

```
/root/ivp-coe/openshift-ansible/roles/openshift_node_dnsmasq/templates/
origin-dns.conf.j2
```

Contents

```
no-resolv
domain-needed
```

```
server=/{{ openshift.common.dns_domain }}/{{ openshift.common.kube_svc_ip }}
no-negcache
max-cache-ttl=60
server=/cmtlab-dns.com/172.22.102.56
```

Verifying Node Accessibility

To verify the reachability of all of the nodes prior to running the installation, you should ping all of the nodes in the cluster from the deployer node.

Running the Ansible Playbook

Once the inventory file has been updated, you run the Ansible playbook. The process shares the login key with all of the nodes so that the deployer node can securely push all of the packages onto the specified nodes. The Ansible playbook process configures the NTP functionality and installs all OpenShift related applications onto the cluster nodes.

Step 1 Within the Linux command line, navigate to the `ivp-coe` directory:

```
cd /root/ivp-coe/
```

Step 2 Execute the following command:

```
ansible-playbook -i abr2ts-inventory share_ssh_key.yml -u root -k
```

The parameters for this command are as follows:

Parameter	Description
`-i`	specifies that this action uses the inventory file
`share_ssh_key.yml`	specifies the keyfile
`-u root`	runs the command as user root
`-k`	prompts for the password

Note The password for all of the nodes is configured within the OVA file and should be identical across the cluster. Please consult with your Cisco representative to securely receive the OVA password.

Sample Output

```
SSH password:

PLAY [OSEv3:deployer]
****************************************************************************************
*************************************************************

TASK [Gathering Facts]
****************************************************************************************
*************************************************************
Sunday 28 January 2018  08:33:39 +0000 (0:00:00.538)       0:00:00.538 ********
ok: [cmt-worker3]
```

```
ok: [cmt-worker1]
ok: [cmt-worker2]
ok: [cmt-master2]
ok: [cmt-infra3]
ok: [cmt-master1]
ok: [cmt-lb1]
ok: [cmt-lb2]
ok: [cmt-master3]
ok: [cmt-infra2]
ok: [cmt-deployer]
ok: [cmt-infra1]

TASK [Add authorized_key on the remote client machine(s)]
*********************************************************************************************
**********************
Sunday 28 January 2018  08:33:41 +0000 (0:00:01.878)       0:00:02.416 ********
changed: [cmt-worker1]
changed: [cmt-worker2]
changed: [cmt-worker3]
changed: [cmt-master2]
changed: [cmt-infra1]
changed: [cmt-master1]
changed: [cmt-infra3]
changed: [cmt-lb2]
changed: [cmt-infra2]
changed: [cmt-deployer]
changed: [cmt-lb1]
changed: [cmt-master3]

PLAY RECAP
*********************************************************************************************
*********************************************************************
cmt-deployer               : ok=2    changed=1    unreachable=0    failed=0
cmt-infra1                 : ok=2    changed=1    unreachable=0    failed=0
cmt-infra2                 : ok=2    changed=1    unreachable=0    failed=0
cmt-infra3                 : ok=2    changed=1    unreachable=0    failed=0
cmt-lb1                    : ok=2    changed=1    unreachable=0    failed=0
cmt-lb2                    : ok=2    changed=1    unreachable=0    failed=0
cmt-master1                : ok=2    changed=1    unreachable=0    failed=0
cmt-master2                : ok=2    changed=1    unreachable=0    failed=0
cmt-master3                : ok=2    changed=1    unreachable=0    failed=0
cmt-worker1                : ok=2    changed=1    unreachable=0    failed=0
cmt-worker2                : ok=2    changed=1    unreachable=0    failed=0
cmt-worker3                : ok=2    changed=1    unreachable=0    failed=0

Sunday 28 January 2018  08:33:41 +0000 (0:00:00.596)       0:00:03.013 ********
===============================================================================
Gathering Facts ---------------------------------------------------------- 1.88s
Add authorized_key on the remote client machine(s) ----------------------- 0.60s
[root@platform ivp-coe]#
```

Performing the Installation

The following steps detail how to run the CMT installation script.

Step 1 Within the Linux command line, navigate to the `ivp-coe` directory:

```
cd /root/ivp-coe/
```

Step 2 Execute the following command:

```
./run_installation install abr2ts-inventory
```

This process will take approximately an hour (with 4 nodes) and will provide feedback as it progresses. A sample of the console output is shown here for your reference. Some highlights will be noted inline.

Sample Output

```
20180128083751|configuration files ...

datestamp=201708181627
yum_repo_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/openshift-1.5.1-internal-yumrep
o_20170720000000_08500bcc3420ab3dee39e9974f8020495b57b000.tar.gz
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/hello-openshift_20170331081125
_762cd2ff99c072584fb8a891cba6b757ab0511b5.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-deployer_20170314105620
_4ed2660e4b9710f083a7a4b907fe849ae92e50df.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-haproxy-router_20170314
105701_38e5ae6da505fcbf3a137e16d5b7b9bbceb612b1.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-keepalived-ipfailover_2
0170403095443_a8c5a2263826f4719a457b8c22ba6985b2c024b7.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-metrics-cassandra_20170
630000000_1d48f1586eaaceea0436cec9a4012bd90f55a1b4.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-metrics-deployer_201706
30000000_0c93d0f3d046f293e8e6b864f4c5c26beb883df7.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-metrics-hawkular-metric
s_20170630000000_f0e0b49a349cd05e5e37ce8744f23295b65b1fbe.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-metrics-heapster_201706
30000000_b2d2724971bf9c9f2f830e122aef432b9795f0f5.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-pod_20170314105448_350f
4bce901ff3baf69f1798eb3f1e411bb58a41.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/hello-openshift_20170619000000
_938808d3d677e42c9156ce6b207cef98ddfbf37f.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-deployer_20170619000000
_6dc526cfb4f95958822485063a79d311e94f015e.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-docker-registry_2017061
9000000_76b578f73ff2fd9132185ab476cae20ea37cbe69.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-egress-router_201706190
00000_22727896e0e7836fb64d0d63c81ca807e29ac287.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-haproxy-router_20170619
000000_fa6ac2db17af2e6d215c09a546cf56a8f50ecbd6.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-keepalived-ipfailover_2
0170619000000_a7a22122945ba905a8f2b190a9bb6e90269b7f52.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-logging-auth-proxy_2017
0619000000_5fe5d013ebff2b8663d7cdd38a7aed068f8e6197.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-logging-curator_2017061
9000000_0f382a3534e143126ce05e61751cd700f42e0e68.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-logging-deployer_201706
19000000_f4ec6b4ac21bece797fde9e13a42e38f23b0c8d6.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-logging-deployment_2017
0619000000_09509eae2df26ccfff40a95ea2ceb22e93d48935.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-logging-elasticsearch_2
0170619000000_685e3e5ae8d20816e91f25b0800e4d6124081544.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-logging-fluentd_2017061
9000000_e6a6da35d74305e47a897a504e8f90ae2285ea65.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-logging-kibana_20170619
000000_18799c4f425e9845416a20536fa558623e069394.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-metrics-cassandra_20170
619000000_c53fdb9f1ee27fbecd3842a5866b6b3c24cbfa49.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-metrics-deployer_201706
19000000_76ed5c3f2becae7396f2f766984bd4085867fcdb.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-metrics-hawkular-metric
s_20170619000000_38afe1c8fc99592cfe2a3e58693c17a88de448bb.tar
```

```
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-metrics-heapster_201706
19000000_8756a4dae71dde92b0e801c1326f2302383f6b7e.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-pod_20170619000000_7fee
59a9167fcb084d5e54eca727480b36988c46.tar
container_file=http://engci-maven-master.cisco.com/artifactory/spvss-ivp-group/origin-recycler_20170619000000
_43c731f410757126c1ca40659582aa793c1ac4c5.tar
coe_gitrepo=https://wwwin-github.cisco.com/spvss-ivp/ivp-coe.git
coe_branch=openshift-1.5.1
openshift-ansible_branch=openshift-ansible-3.5.116-1
coe_git_commit_full=4b9c2c44f5ca6a4c48308311a7d5debdb7d4b5cb
coe_git_commit_short=4b9c2c4
……..
TASK [/root/ivp-coe/openshift-ansible/roles/os_firewall : Start and enable iptables service]
*********************************************************************
Sunday 28 January 2018  09:36:03 +0000 (0:00:00.862)       0:00:26.319 ********
ok: [cmt-worker2]
ok: [cmt-worker3]
ok: [cmt-worker1]
ok: [cmt-infra1]
ok: [cmt-infra2]
ok: [cmt-infra3]

TASK [/root/ivp-coe/openshift-ansible/roles/os_firewall : need to pause here, otherwise the iptables service
starting can sometimes cause ssh to fail] *********
Sunday 28 January 2018  09:36:04 +0000 (0:00:00.390)       0:00:26.710 ********
skipping: [cmt-worker1]

TASK [/root/ivp-coe/openshift-ansible/roles/os_firewall : Add iptables allow rules]
*********************************************************************************
Sunday 28 January 2018  09:36:04 +0000 (0:00:00.050)       0:00:26.761 ********
TASK [/root/ivp-coe/openshift-ansible/roles/os_firewall : Remove iptables rules]
*********************************************************************************
Sunday 28 January 2018  09:36:04 +0000 (0:00:00.087)       0:00:26.848 ********
```

Note Just below, is a “PLAY RECAP” section that shows the installed nodes and will display a “failed=0” status if the installation was successful.

```
PLAY RECAP
*********************************************************************************************************************
*************************************
cmt-infra1                 : ok=15   changed=3    unreachable=0    failed=0
cmt-infra2                 : ok=15   changed=3    unreachable=0    failed=0
cmt-infra3                 : ok=15   changed=3    unreachable=0    failed=0
cmt-lb1                    : ok=2    changed=1    unreachable=0    failed=0
cmt-lb2                    : ok=2    changed=1    unreachable=0    failed=0
cmt-master1                : ok=3    changed=2    unreachable=0    failed=0
cmt-master2                : ok=2    changed=2    unreachable=0    failed=0
cmt-master3                : ok=2    changed=2    unreachable=0    failed=0
cmt-worker1                : ok=15   changed=3    unreachable=0    failed=0
cmt-worker2                : ok=15   changed=3    unreachable=0    failed=0
cmt-worker3                : ok=15   changed=3    unreachable=0    failed=0
localhost                  : ok=3    changed=3    unreachable=0    failed=0

Sunday 28 January 2018  09:36:04 +0000 (0:00:00.092)       0:00:26.941 ********
===============================================================================
seboolean ------------------------------------------------------------ 4.84s
post-installation | configure ipfailover | oc label nodes --------------- 3.76s
openshift_facts : Gather Cluster facts and set is_containerized if needed --- 3.20s
openshift_facts : Ensure various deps are installed ---------------------- 2.64s
post-installation | cluster configuration | oc adm policy add-cluster-role-to-user cluster-admin system ---
2.34s
```

```
post-installation | cluster configuration | oc edit scc restricted ------ 1.48s
restart origin-node ---------------------------------------------------- 1.38s
Gathering Facts -------------------------------------------------------- 1.06s
……..

……
TASK [gluster | execute template]
*********************************************************************************************************
*****************
Sunday 28 January 2018  09:36:05 +0000 (0:00:00.030)       0:00:00.152 ********
skipping: [localhost] => (item=oc create -f /tmp/gluster_template.yml)

PLAY RECAP
*********************************************************************************************************
*************************************
localhost                  : ok=0    changed=0    unreachable=0    failed=0

Sunday 28 January 2018  09:36:05 +0000 (0:00:00.029)       0:00:00.181 ********
===============================================================================
gluster | generate template ---------------------------------------------- 0.03s
gluster | execute template ----------------------------------------------- 0.03s
20180128093605|install complete ... starting containers

20180128093605|waiting for router deployer to complete ...
20180128093705|done

20180128093705|deployment complete
```

Note Below is the hostname (or IP address) for the load balancer VIP that we configured. If you are using a hostname, it should be configured in the `/etc/hosts` file on the Deployer node.

```
In project default on server https://cmt-osp-cluster.cmtlab-dns.com:8443

svc/kubernetes - 172.30.0.1 ports 443, 53->8053, 53->8053

View details with 'oc describe <resource>/<name>' or list everything with 'oc get all'.
```

Note Notice the listing of the nodes, with the statuses shown. All nodes should be in a "Ready" state with the master nodes being "Ready,SchedulingDisabled".

```
NAME          STATUS                     AGE         EXTERNAL-IP
cmt-infra1    Ready                      3m          <none>
cmt-infra2    Ready                      3m          <none>
cmt-infra3    Ready                      3m          <none>
cmt-master1   Ready,SchedulingDisabled   3m          <none>
cmt-master2   Ready,SchedulingDisabled   3m          <none>
cmt-master3   Ready,SchedulingDisabled   3m          <none>
cmt-worker1   Ready                      3m          <none>
cmt-worker2   Ready                      3m          <none>
cmt-worker3   Ready                      3m          <none>

NAME             CLUSTER-IP   EXTERNAL-IP   PORT(S)                  AGE
svc/kubernetes   172.30.0.1   <none>        443/TCP,53/UDP,53/TCP 7h
```

Note Lastly, at the end of the output, there will be an "installation finished" confirmation if the installation process was successful.

```
20180128093706|installation finished
20180128093706|RC=0|MSG=exiting
```

Verifying the Installation

The following section details the approaches to validating various aspects of the CMT installation.

OpenShift Verification Commands

Command Line Verification

There are a number of command line options for verifying the integrity of your CMT deployment and for checking the current status of nodes, connections, and services.

To verify a successful installation from the command line, type the following command:

```
oc login -u system -p admin --insecure-skip-tls-verify=true -n default
https://cmt-osp-cluster.cmtlab-dns.com:8443
```

The parameters for this command are as follows:

Parameter	Description
-u	username = system
-p	password = admin
--insecure-skip-tls	logs in insecurely (not using tls)
verify	true to verify the project integrity
-n	Switches to the “default” project.

Sample output is as follows:

```
Login successful.

You have access to the following projects and can switch between them with ‘oc project
<projectname>’:
    *default
    kube-system
    logging
    management-infra
    openshift
    openshift-infra

Using project “default”.
Welcome! See 'oc help' to get started.
```

The following command verifies that you can login to the default OpenShift instance using an IP address:

Command

```
oc login -u system -p admin --insecure-skip-tls-verify=true -n default
https://172.22.102.244:8443
```

Output

```
Login successful.

You have access to the following projects and can switch between them with 'oc project
<projectname>':

  * default
    kube-system
    logging
    management-infra
    openshift
    openshift-infra

Using project "default".
Welcome! See 'oc help' to get started.
[root@cmt-deployer ~]# oc get nodes
NAME          STATUS                     AGE
cmt-infra1    Ready                      4h
cmt-infra2    Ready                      4h
cmt-infra3    Ready                      4h
cmt-master1   Ready,SchedulingDisabled   4h
cmt-master2   Ready,SchedulingDisabled   4h
cmt-master3   Ready,SchedulingDisabled   4h
cmt-worker1   Ready                      4h
cmt-worker2   Ready                      4h
cmt-worker3   Ready                      4h
[root@cmt-deployer ~]#
```

The following command verifies that you can login to the default OpenShift instance using a hostname:

Command

```
[root@cmt-deployer ~]# oc login -u system -p admin --insecure-skip-tls-verify=true -n
default https://cmt-osp-cluster.cmtlab-dns.com:8443
```

Output

```
Login successful.

You have access to the following projects and can switch between them with 'oc project
<projectname>':

  * default
    kube-system
    logging
    management-infra
    openshift
    openshift-infra

Using project "default".
Welcome! See 'oc help' to get started.
[root@cmt-deployer ~]# oc get nodes
NAME          STATUS                     AGE
cmt-infra1    Ready                      4h
cmt-infra2    Ready                      4h
cmt-infra3    Ready                      4h
```

```
cmt-master1   Ready,SchedulingDisabled   4h
cmt-master2   Ready,SchedulingDisabled   4h
cmt-master3   Ready,SchedulingDisabled   4h
cmt-worker1   Ready                      4h
cmt-worker2   Ready                      4h
cmt-worker3   Ready                      4h
[root@cmt-deployer ~]#
```

The following commands can be used to verify the status of OpenShift. The first command lists available nodes and how long they have been running.

Command

```
[root@platform ivp-coe]# oc get nodes
```

Output

```
NAME          STATUS                     AGE
cmt-infra1    Ready                      4h
cmt-infra2    Ready                      4h
cmt-infra3    Ready                      4h
cmt-master1   Ready,SchedulingDisabled   4h
cmt-master2   Ready,SchedulingDisabled   4h
cmt-master3   Ready,SchedulingDisabled   4h
cmt-worker1   Ready                      4h
cmt-worker2   Ready                      4h
cmt-worker3   Ready                      4h
```

The `oc status` command tells you which project you are in and which services are running.

Command

```
[root@platform ivp-coe]# oc status
```

Output:

```
In project default on server https://cmt-osp-cluster.cmtlab-dns.com:8443

svc/kubernetes - 172.30.0.1 ports 443, 53->8053, 53->8053

View details with 'oc describe <resource>/<name>' or list everything with 'oc get all'.
```

Note When the oc status command is run, it should not return any errors

Command: `[root@platform ivp-coe]# oc get all`

Output:

```
NAME             CLUSTER-IP   EXTERNAL-IP   PORT(S)                  AGE
svc/kubernetes   172.30.0.1   <none>        443/TCP,53/UDP,53/TCP   23m
```

Verifying the NIC & Node Labels

Next, you must run a command on the deployer node to show all of the NIC and labels for all of the nodes.

Worker nodes will be labeled as `cisco.com/type=backend`, while infra nodes will be labeled as `cisco.com/type=master` and `infra.cisco.com/type=infra`

```
[root@cmt-deployer ~]# oc get nodes --show-labels
```

Sample Output

```
NAME           STATUS                     AGE       LABELS
cmt-infra1     Ready                      4h
beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,cisco.com/type=master,infra.cisco.com/type=infra,ku
bernetes.io/hostname=cmt-infra1,network.cisco.com/eth0=172.22.102.58,network.cisco.com/eth1=192.169.131.5,net
work.cisco.com/lo=127.0.0.1,region=infra
cmt-infra2     Ready                      4h
beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,cisco.com/type=infra,infra.cisco.com/type=infra,kub
ernetes.io/hostname=cmt-infra2,network.cisco.com/eth0=172.22.102.61,network.cisco.com/eth1=192.169.131.6,netw
ork.cisco.com/lo=127.0.0.1,region=infra
cmt-infra3     Ready                      4h
beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,cisco.com/type=master,infra.cisco.com/type=infra,ku
bernetes.io/hostname=cmt-infra3,network.cisco.com/eth0=172.22.102.65,network.cisco.com/eth1=192.169.131.7,net
work.cisco.com/lo=127.0.0.1,region=infra
cmt-master1    Ready,SchedulingDisabled   4h
beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,kubernetes.io/hostname=cmt-master1
cmt-master2    Ready,SchedulingDisabled   4h
beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,kubernetes.io/hostname=cmt-master2
cmt-master3    Ready,SchedulingDisabled   4h
beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,kubernetes.io/hostname=cmt-master3
cmt-worker1    Ready                      4h
beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,cisco.com/type=backend,kubernetes.io/hostname=cmt-w
orker1,network.cisco.com/eth0=172.22.102.152,network.cisco.com/eth1=192.169.131.2,network.cisco.com/lo=127.0.
0.1,region=infra
cmt-worker2    Ready                      4h
beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,cisco.com/type=backend,kubernetes.io/hostname=cmt-w
orker2,network.cisco.com/eth0=172.22.102.153,network.cisco.com/eth1=192.169.131.3,network.cisco.com/lo=127.0.
0.1,region=infra
cmt-worker3    Ready                      4h
beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,cisco.com/type=backend,kubernetes.io/hostname=cmt-w
orker3,network.cisco.com/eth0=172.22.102.250,network.cisco.com/eth1=192.169.131.4,network.cisco.com/lo=127.0.
0.1,region=infra
[root@cmt-deployer ~]#
```

GUI Verification

Once you have completed the installation you will need to configure GUI access.

Step 1 Add the `<LB VIP> <domain-name>` to your local machine on `/etc/hosts/`.

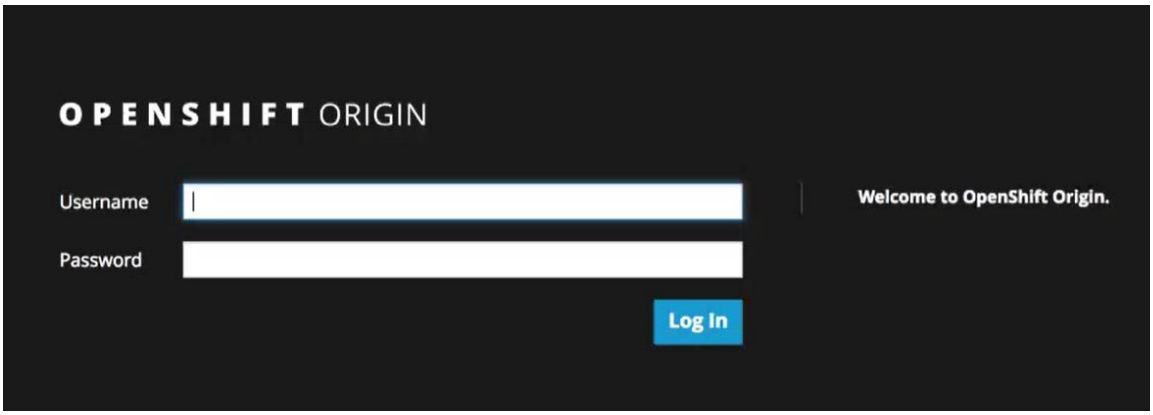
For example: `172.22.102.244  cmt-osp-cluster.cmtlab-dns.com`

Step 2 You should be able to access the OpenShift cluster console for CMT by appending port 8443 to the address of the Virtual IP as follows.

`https://cmt-osp-cluster.cmtlab-dns.com:8443`

Step 3 The OpenShift login console should appear. Default credentials are `user: system` / `password: admin`, but you should immediately change them to unique, secure credentials of your own choosing.

Figure 3-1 OpenShift Origin Login Screen



Step 4 Once you have logged in, you should see a list of existing projects.

Updating the Cluster Port Range

The following command allows CMT to be accessible via port 80. That port is disabled by default, as the OpenShift infrastructure uses it. To enable port 80:

Step 1 Navigate to the following directory:

```
cd /root/ivp-coe/vmr/
```

Step 2 Edit `dp_mods.yml` to update the servicesNodePortRange value as shown below:

```
line: '\1servicesNodePortRange: "80-9999"'
```

Step 3 Navigate to the following directory:

```
cd /root/ivp-coe/
```

Step 4 Run the following command to enable port 80. The process will take approximately 2 minutes to complete:

```
[root@cmt-deployer ivp-coe]# ansible-playbook -i abr2ts-inventory vmr/dp_mods.yml
```

Output

```
PLAY [selinux]
*************************************************************************************
*********************************************************************

TASK [Gathering Facts]
*************************************************************************************
*********************************************************
Sunday 28 January 2018  10:29:23 +0000 (0:00:00.120)       0:00:00.120 ********
ok: [cmt-worker1]
ok: [cmt-master1]
ok: [cmt-master2]
ok: [cmt-worker2]
ok: [cmt-worker3]
ok: [cmt-infra1]
ok: [cmt-master3]
```

```
ok: [cmt-infra3]
ok: [cmt-infra2]

TASK [SELINUX State]
*********************************************************************************************
************************************************************
Sunday 28 January 2018  10:29:24 +0000 (0:00:01.820)       0:00:01.941 ********
changed: [cmt-master2]
changed: [cmt-worker2]
changed: [cmt-worker3]
changed: [cmt-worker1]
changed: [cmt-master3]
changed: [cmt-infra1]
changed: [cmt-infra3]
changed: [cmt-infra2]
changed: [cmt-master1]

PLAY [Unsecure OS]
*********************************************************************************************
************************************************************

TASK [stat]
*********************************************************************************************
*******************************************************************
Sunday 28 January 2018  10:29:25 +0000 (0:00:00.415)       0:00:02.356 ********
ok: [cmt-master1 -> localhost]

TASK [set_fact]
*********************************************************************************************
************************************************************
Sunday 28 January 2018  10:29:25 +0000 (0:00:00.366)       0:00:02.723 ********
skipping: [cmt-master1]

TASK [set_fact]
*********************************************************************************************
************************************************************
Sunday 28 January 2018  10:29:25 +0000 (0:00:00.036)       0:00:02.760 ********
skipping: [cmt-master1]

TASK [Copy OC Apply file]
*********************************************************************************************
*****************************************************
Sunday 28 January 2018  10:29:25 +0000 (0:00:00.033)       0:00:02.794 ********
changed: [cmt-master1]

TASK [OC Apply]
*********************************************************************************************
************************************************************
Sunday 28 January 2018  10:29:26 +0000 (0:00:00.672)       0:00:03.466 ********
changed: [cmt-master1]

PLAY [Unsecure OS]
*********************************************************************************************
************************************************************

TASK [Check if Service Exists]
*********************************************************************************************
***********************************************
Sunday 28 January 2018  10:29:27 +0000 (0:00:01.155)       0:00:04.621 ********
ok: [cmt-master1]
ok: [cmt-master2]
ok: [cmt-master3]
```

```
TASK [set_fact]
*******************************************************************************************
*****************************************************************
Sunday 28 January 2018  10:29:27 +0000 (0:00:00.287)       0:00:04.909 ********
skipping: [cmt-master1]
skipping: [cmt-master2]
skipping: [cmt-master3]

TASK [set_fact]
*******************************************************************************************
*****************************************************************
Sunday 28 January 2018  10:29:27 +0000 (0:00:00.055)       0:00:04.964 ********
ok: [cmt-master1]
ok: [cmt-master2]
ok: [cmt-master3]

TASK [Port Range Change]
*******************************************************************************************
*****************************************************
Sunday 28 January 2018  10:29:28 +0000 (0:00:00.066)       0:00:05.031 ********
changed: [cmt-master1]
changed: [cmt-master2]
changed: [cmt-master3]

RUNNING HANDLER [restart_origin_master_multi]
*******************************************************************************************
**********************************
Sunday 28 January 2018  10:29:28 +0000 (0:00:00.349)       0:00:05.381 ********
changed: [cmt-master2] => (item=origin-master-controllers)
changed: [cmt-master1] => (item=origin-master-controllers)
changed: [cmt-master3] => (item=origin-master-controllers)
changed: [cmt-master3] => (item=origin-master-api)
changed: [cmt-master1] => (item=origin-master-api)
changed: [cmt-master2] => (item=origin-master-api)

PLAY RECAP
*******************************************************************************************
*******************************************************************
cmt-infra1                 : ok=2    changed=1    unreachable=0    failed=0
cmt-infra2                 : ok=2    changed=1    unreachable=0    failed=0
cmt-infra3                 : ok=2    changed=1    unreachable=0    failed=0
cmt-master1                : ok=9    changed=5    unreachable=0    failed=0
cmt-master2                : ok=6    changed=3    unreachable=0    failed=0
cmt-master3                : ok=6    changed=3    unreachable=0    failed=0
cmt-worker1                : ok=2    changed=1    unreachable=0    failed=0
cmt-worker2                : ok=2    changed=1    unreachable=0    failed=0
cmt-worker3                : ok=2    changed=1    unreachable=0    failed=0

Sunday 28 January 2018  10:29:30 +0000 (0:00:01.895)       0:00:07.276 ********
===============================================================================
restart_origin_master_multi ------------------------------------------- 1.90s
Gathering Facts ------------------------------------------------------- 1.82s
OC Apply -------------------------------------------------------------- 1.16s
Copy OC Apply file ---------------------------------------------------- 0.67s
SELINUX State --------------------------------------------------------- 0.42s
stat ------------------------------------------------------------------ 0.37s
Port Range Change ----------------------------------------------------- 0.35s
Check if Service Exists ----------------------------------------------- 0.29s
set_fact -------------------------------------------------------------- 0.07s
set_fact -------------------------------------------------------------- 0.06s
set_fact -------------------------------------------------------------- 0.04s
set_fact -------------------------------------------------------------- 0.03s
[root@cmt-deployer ivp-coe]#
```

Updating iptables

Next, you will need to update the iptables for the Infra1 and Infra3 nodes using the following procedure:

Step 1 SSH into the Infra1 node.

Step 2 Check the iptable rules with the following command:

```
[root@cmt-infra1 ~]# iptables -L OS_FIREWALL_ALLOW -n --line-numbers
```

Output

```
Chain OS_FIREWALL_ALLOW (1 references)
num  target     prot opt source               destination
1    ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:123
2    ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:10250
3    ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:80
4    ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:443
5    ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:10255
6    ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:10255
7    ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:4789
```

Step 3 This command sets the iptable rule.

```
[root@cmt-infra1 ~]# iptables -R OS_FIREWALL_ALLOW 3 -m tcp -p tcp --dport 80 -j ACCEPT
```

Step 4 Now we verify that the applied rule exists within the iptable. Note that port 80 access is now shown as enabled in the output below.

```
[root@cmt-infra1 ~]# iptables -L OS_FIREWALL_ALLOW -n --line-numbers
```

Output

```
Chain OS_FIREWALL_ALLOW (1 references)
num  target     prot opt source               destination
1    ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:123
2    ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:10250
3    ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:80
4    ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:443
5    ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:10255
6    ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:10255
7    ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:4789
```

Step 5 Repeat this entire process with the Infra3 node.

Configuring the IPVS VIP on all Worker Nodes

To start, a new IP address must be provided for the IPVS VIP. That address must be available on the Eth1: network. To allocate this address:

Step 1 SSH into the deployer node as root.

Step 2 Change into the “ivp-coe” directory.

```
[root@cmt-deployer ~]# cd /root/ivp-coe/
```

Step 3 Copy and overwrite the `yaml` file.

```
[root@cmt-deployer ivp-coe]# cp
/root/abr2ts-deployment/scripts/add_ip_to_interface.yaml  contrib/
```

Step 4 Confirm the file overwrite.

```
cp: overwrite 'contrib/add_ip_to_interface.yaml'? y
```

Step 5 Run the following command and update the IP addresses (in bold) to match your IPVS VIP.

For easier readability, the arguments are placed on separate lines here:

Command

```
[root@cmt-deployer ivp-coe]#
ansible-playbook -i abr2ts-inventory ./contrib/add_ip_to_interface.yaml
-e "node_selector_label='cisco.com/type: backend'"
-e "interface=lo"
-e "address=192.169.131.1"
-e "netmask=255.255.255.255"
-e "network=192.169.131.0"
-e "broadcast=192.169.131.255"
```

Step 6 The command will take approximately 5 minutes to run. In the output, look at the PLAY RECAP at the end, you will see a “failed = 0” if the process was successful.

Output

```
PLAY [contrib | add_ip_to_interface]
*********************************************************************************************
*****************************************

TASK [Gathering Facts]
*********************************************************************************************
*********************************************************
Sunday 28 January 2018  11:13:26 +0000 (0:00:00.117)       0:00:00.117 ********
ok: [cmt-infra1]
ok: [cmt-worker2]
ok: [cmt-worker1]
ok: [cmt-infra2]
ok: [cmt-infra3]
ok: [cmt-worker3]

TASK [contrib | add_ip_to_interface | check variables are set]
*********************************************************************************************
******************
Sunday 28 January 2018  11:13:28 +0000 (0:00:02.246)       0:00:02.363 ********
ok: [cmt-worker2] => (item=node_selector_label) => {
    "changed": false,
    "item": "node_selector_label",
    "msg": "All assertions passed"
}
ok: [cmt-worker1] => (item=interface) => {
    "changed": false,
    "item": "interface",
    "msg": "All assertions passed"
}
ok: [cmt-worker1] => (item=node_selector_label) => {
    "changed": false,
    "item": "node_selector_label",
    "msg": "All assertions passed"
}
ok: [cmt-worker3] => (item=broadcast) => {
    "changed": false,
    "item": "broadcast",
    "msg": "All assertions passed"
}
ok: [cmt-worker3] => (item=network) => {
    "changed": false,
```

```
    "item": "network",
    "msg": "All assertions passed"
}
ok: [cmt-worker3] => (item=netmask) => {
    "changed": false,
    "item": "netmask",
    "msg": "All assertions passed"
}
ok: [cmt-worker3] => (item=address) => {
    "changed": false,
    "item": "address",
    "msg": "All assertions passed"
}
ok: [cmt-worker3] => (item=interface) => {
    "changed": false,
    "item": "interface",
    "msg": "All assertions passed"
}
ok: [cmt-worker3] => (item=node_selector_label) => {
    "changed": false,
    "item": "node_selector_label",
    "msg": "All assertions passed"
}
ok: [cmt-worker1] => (item=broadcast) => {
    "changed": false,
    "item": "broadcast",
    "msg": "All assertions passed"
}
ok: [cmt-worker1] => (item=network) => {
    "changed": false,
    "item": "network",
    "msg": "All assertions passed"
}
ok: [cmt-worker1] => (item=netmask) => {
    "changed": false,
    "item": "netmask",
    "msg": "All assertions passed"
}
ok: [cmt-worker1] => (item=address) => {
    "changed": false,
    "item": "address",
    "msg": "All assertions passed"
}
ok: [cmt-worker2] => (item=broadcast) => {
    "changed": false,
    "item": "broadcast",
    "msg": "All assertions passed"
}
ok: [cmt-worker2] => (item=network) => {
    "changed": false,
    "item": "network",
    "msg": "All assertions passed"
}
ok: [cmt-worker2] => (item=netmask) => {
    "changed": false,
    "item": "netmask",
    "msg": "All assertions passed"
}
ok: [cmt-worker2] => (item=address) => {
    "changed": false,
    "item": "address",
    "msg": "All assertions passed"
}
ok: [cmt-worker2] => (item=interface) => {
```

```
    "changed": false,
    "item": "interface",
    "msg": "All assertions passed"
}
ok: [cmt-infra2] => (item=interface) => {
    "changed": false,
    "item": "interface",
    "msg": "All assertions passed"
}
ok: [cmt-infra2] => (item=node_selector_label) => {
    "changed": false,
    "item": "node_selector_label",
    "msg": "All assertions passed"
}
ok: [cmt-infra1] => (item=node_selector_label) => {
    "changed": false,
    "item": "node_selector_label",
    "msg": "All assertions passed"
}
ok: [cmt-infra2] => (item=address) => {
    "changed": false,
    "item": "address",
    "msg": "All assertions passed"
}
ok: [cmt-infra1] => (item=interface) => {
    "changed": false,
    "item": "interface",
    "msg": "All assertions passed"
}
ok: [cmt-infra1] => (item=address) => {
    "changed": false,
    "item": "address",
    "msg": "All assertions passed"
}
ok: [cmt-infra1] => (item=netmask) => {
    "changed": false,
    "item": "netmask",
    "msg": "All assertions passed"
}
ok: [cmt-infra3] => (item=node_selector_label) => {
    "changed": false,
    "item": "node_selector_label",
    "msg": "All assertions passed"
}
ok: [cmt-infra3] => (item=interface) => {
    "changed": false,
    "item": "interface",
    "msg": "All assertions passed"
}
ok: [cmt-infra3] => (item=address) => {
    "changed": false,
    "item": "address",
    "msg": "All assertions passed"
}
ok: [cmt-infra2] => (item=network) => {
    "changed": false,
    "item": "network",
    "msg": "All assertions passed"
}
ok: [cmt-infra2] => (item=netmask) => {
    "changed": false,
    "item": "netmask",
    "msg": "All assertions passed"
}
```

```
ok: [cmt-infra1] => (item=broadcast) => {
    "changed": false,
    "item": "broadcast",
    "msg": "All assertions passed"
}
ok: [cmt-infra1] => (item=network) => {
    "changed": false,
    "item": "network",
    "msg": "All assertions passed"
}
ok: [cmt-infra2] => (item=broadcast) => {
    "changed": false,
    "item": "broadcast",
    "msg": "All assertions passed"
}
ok: [cmt-infra3] => (item=netmask) => {
    "changed": false,
    "item": "netmask",
    "msg": "All assertions passed"
}
ok: [cmt-infra3] => (item=network) => {
    "changed": false,
    "item": "network",
    "msg": "All assertions passed"
}
ok: [cmt-infra3] => (item=broadcast) => {
    "changed": false,
    "item": "broadcast",
    "msg": "All assertions passed"
}

TASK [set_fact]
*******************************************************************************************
*****************************************************************
Sunday 28 January 2018  11:13:29 +0000 (0:00:00.163)       0:00:02.527 ********
ok: [cmt-worker1]
ok: [cmt-worker3]
ok: [cmt-infra1]
ok: [cmt-worker2]
ok: [cmt-infra2]
ok: [cmt-infra3]

TASK [contrib | add_ip_to_interface | add template file]
*******************************************************************************************
************************
Sunday 28 January 2018  11:13:29 +0000 (0:00:00.094)       0:00:02.621 ********
skipping: [cmt-infra1]
skipping: [cmt-infra2]
skipping: [cmt-infra3]
changed: [cmt-worker2]
changed: [cmt-worker1]
changed: [cmt-worker3]

TASK [contrib | add_ip_to_interface | configure sysctl]
*******************************************************************************************
************************
Sunday 28 January 2018  11:13:30 +0000 (0:00:00.903)       0:00:03.525 ********
skipping: [cmt-infra1]
skipping: [cmt-infra2]
skipping: [cmt-infra3]
changed: [cmt-worker3]
changed: [cmt-worker2]
changed: [cmt-worker1]
```

```
TASK [contrib | add_ip_to_interface | configure sysctl]
*********************************************************************************************
************************
Sunday 28 January 2018  11:13:30 +0000 (0:00:00.418)       0:00:03.943 ********
skipping: [cmt-infra1]
skipping: [cmt-infra2]
skipping: [cmt-infra3]
changed: [cmt-worker2]
changed: [cmt-worker1]
changed: [cmt-worker3]

RUNNING HANDLER [restart-network]
*********************************************************************************************
*******************************************
Sunday 28 January 2018  11:13:30 +0000 (0:00:00.243)       0:00:04.187 ********
changed: [cmt-worker1]
changed: [cmt-worker3]
changed: [cmt-worker2]

PLAY RECAP
*********************************************************************************************
*********************************************************************
cmt-infra1                 : ok=3    changed=0    unreachable=0    failed=0
cmt-infra2                 : ok=3    changed=0    unreachable=0    failed=0
cmt-infra3                 : ok=3    changed=0    unreachable=0    failed=0
cmt-worker1                : ok=7    changed=4    unreachable=0    failed=0
cmt-worker2                : ok=7    changed=4    unreachable=0    failed=0
cmt-worker3                : ok=7    changed=4    unreachable=0    failed=0

Sunday 28 January 2018  11:13:33 +0000 (0:00:02.283)       0:00:06.470 ********
===============================================================================
restart-network ------------------------------------------------------- 2.28s
Gathering Facts ------------------------------------------------------- 2.25s
contrib | add_ip_to_interface | add template file ----------------------- 0.90s
contrib | add_ip_to_interface | configure sysctl ------------------------ 0.42s
contrib | add_ip_to_interface | configure sysctl ------------------------ 0.24s
contrib | add_ip_to_interface | check variables are set ----------------- 0.16s
set_fact -------------------------------------------------------------- 0.09s
[root@cmt-deployer ivp-coe]#
```

Verifying the IPVS VIP on all Worker Nodes

To verify that the IPVS VIP has been properly added to the lo:1 interfaces, execute the following command on each worker node:

```
ip a | grep <VIP>
```

Sample Command & Output

```
[root@cmt-worker3 ~]# ip a | grep 192.168.131.1
    inet 192.169.131.1/32 brd 192.169.131.255 scope global lo:1
```

Load Images into Docker Registry

Next, you will need to load the application images for CMT, IPVS, logging, and monitoring, to the Docker registry. The images reside at the following location on the Deployer node:

```
[root@cmt-deployer images]# /root/abr2ts-deployment/abr2ts-docker-images/images
```

Step 1 If necessary, log into the Deployer node.

```
ssh {username}@{Deployer_Server_IP}
```

The application images are as shown:

```
[root@cmt-deployer abr2ts-docker-images]# ls -ltr
total 4526340
-rw-------. 1 root root 817498112 Jan 26 04:25 fluent.cisco-1.0.0_015.tar.gz
-rw-------. 1 root root 292882944 Jan 26 04:26 grafana.cisco-1.0.0_013.tar.gz
-rw-------. 1 root root 561941504 Jan 26 04:26 ipvs_keepalived.cisco-1.0.0-26.tar
-rw-------. 1 root root 623892992 Jan 26 04:26 kafka-20180110164016-0.10.2.tar.gz
-rw-------. 1 root root 308331008 Jan 26 04:27 kafka-exporter-20180108142414-0.3.0.tar.gz
-rw-------. 1 root root 394202112 Jan 26 04:27
logging-bundle-ansible-bundle-20180110164043-17.4.3.tar
-rw-------. 1 root root 840505856 Jan 26 04:28 logstash-20180108142323-5.5.0.tar.gz
-rw-------. 1 root root  76923392 Jan 26 04:28 prometheus.cisco-1.0.0_013.tar.gz
-rw-------. 1 root root  14503424 Jan 26 04:28 proxytoservice-20180108143454-1.0.0.tar.gz
-rw-------. 1 root root        70 Jan 26 04:28 README.md
-rw-------. 1 root root 672744960 Jan 26 04:28 zookeeper-20180108141946-3.5.2.tar.gz
-rw-------. 1 root root  19234304 Jan 30 03:23 alertmanager.cisco-1.0.0_014.tar.gz
-rw-------. 1 root root  12288000 Jan 30 18:59 vod-gateway.cisco-1.0.0_3.tar.gz
[root@platform images]#
```

Step 2 Change to the scripts directory:

```
cd /root/abr2ts-deployment/scripts
```

Step 3 Run the following command:

```
./load_to_registry.sh {Deployer_Server_IP}
```

for example:

```
[root@cmt-deployer scripts]# ./load_to_registry.sh 172.22.102.170
```

Sample Output

```
[root@cmt-deployer scripts]# ./load_to_registry.sh 172.22.102.170
LOAD Script
cisco-1.0.0_427
docker load -i ../abr2ts-docker-images/images/vod-gateway.cisco-1.0.0_427.tar.gz
7e3694659f4b: Loading layer [==================================================>] 4.223 MB/4.223 MB
f77ddf18cac9: Loading layer [==================================================>] 4.096 kB/4.096 kB
acc75f2075b8: Loading layer [==================================================>] 3.072 kB/3.072 kB
92bbbcd3d1ee: Loading layer [==================================================>]  8.02 MB/8.02 MB
070117c17c8a: Loading layer [==================================================>]  2.56 kB/2.56 kB
2497c06e4230: Loading layer [==================================================>] 3.072 kB/3.072 kB
Loaded image: abr2ts_release/vod-gateway:cisco-1.0.0_427
Prev Tag: abr2ts_release/vod-gateway:cisco-1.0.0_427
Tagging: 172.22.102.170:5000/abr2ts_release/vod-gateway:cisco-1.0.0_427
Docker push: 172.22.102.170:5000/abr2ts_release/vod-gateway:cisco-1.0.0_427
The push refers to a repository [172.22.102.170:5000/abr2ts_release/vod-gateway]
2497c06e4230: Pushed
070117c17c8a: Pushed
92bbbcd3d1ee: Pushed
acc75f2075b8: Pushed
f77ddf18cac9: Pushed
7e3694659f4b: Pushed
cisco-1.0.0_427: digest: sha256:461400611452630b2a18158d433eef02536eedf0be1dcf9a806ea94c76931c4b size: 1567
The push refers to a repository [172.22.102.170:5000/abr2ts_release/vod-gateway]
2497c06e4230: Layer already exists
070117c17c8a: Layer already exists
92bbbcd3d1ee: Layer already exists
```

```
acc75f2075b8: Layer already exists
f77ddf18cac9: Layer already exists
7e3694659f4b: Layer already exists
latest: digest: sha256:461400611452630b2a18158d433eef02536eedf0be1dcf9a806ea94c76931c4b size: 1567
Untagged: abr2ts_release/vod-gateway:cisco-1.0.0_427
Untagged:
172.22.102.170:5000/abr2ts_release/vod-gateway@sha256:461400611452630b2a18158d433eef02536eedf0be1dcf9a806ea94
c76931c4b
cisco-1.0.0_015
docker load -i ../abr2ts-docker-images/images/fluent.cisco-1.0.0_015.tar.gz
78ff13900d61: Loading layer [==================================================>] 196.8 MB/196.8 MB
641fcd2417bc: Loading layer [==================================================>] 209.9 kB/209.9 kB
292a66992f77: Loading layer [==================================================>] 7.168 kB/7.168 kB
3567b2f05514: Loading layer [==================================================>] 4.608 kB/4.608 kB
367b9c52c931: Loading layer [==================================================>] 3.072 kB/3.072 kB
efdf063314e7: Loading layer [==================================================>] 22.26 MB/22.26 MB
3dea68c34942: Loading layer [==================================================>] 275.4 MB/275.4 MB
625d45015bed: Loading layer [==================================================>] 260.6 MB/260.6 MB
d938d6655758: Loading layer [==================================================>]   126 kB/126 kB
5695a7ee01c0: Loading layer [==================================================>] 475.6 kB/475.6 kB
d66bf69bed8d: Loading layer [==================================================>] 1.804 MB/1.804 MB
eb6427fb215b: Loading layer [==================================================>] 2.942 MB/2.942 MB
9f2bba04a565: Loading layer [==================================================>] 3.584 kB/3.584 kB
98d85932a25b: Loading layer [==================================================>] 3.584 kB/3.584 kB
2aff46eaac4c: Loading layer [==================================================>] 3.584 kB/3.584 kB
088302c22591: Loading layer [==================================================>] 5.632 kB/5.632 kB
c3383b41c607: Loading layer [==================================================>] 45.17 MB/45.17 MB
dedb3c7497d1: Loading layer [==================================================>]   151 kB/151 kB
9ef0e71a9476: Loading layer [==================================================>] 300.5 kB/300.5 kB
440537231155: Loading layer [==================================================>] 61.44 kB/61.44 kB
1352fbf1785a: Loading layer [==================================================>] 228.9 kB/228.9 kB
2f924f30f505: Loading layer [==================================================>] 1.254 MB/1.254 MB
8019f73b25da: Loading layer [==================================================>] 239.1 kB/239.1 kB
c21cd402f80f: Loading layer [==================================================>] 9.343 MB/9.343 MB
6b416303629f: Loading layer [==================================================>] 4.096 kB/4.096 kB
15172e48c360: Loading layer [==================================================>] 4.096 kB/4.096 kB
80ff5081dbe2: Loading layer [==================================================>] 4.096 kB/4.096 kB
30b2eeafe84a: Loading layer [==================================================>] 3.584 kB/3.584 kB
3eee1d1f03b4: Loading layer [==================================================>] 3.584 kB/3.584 kB
b33d84e048fb: Loading layer [==================================================>] 4.096 kB/4.096 kB
946c30139423: Loading layer [==================================================>] 5.632 kB/5.632 kB
Loaded image: abr2ts_release/fluent:cisco-1.0.0_015
Prev Tag: abr2ts_release/fluent:cisco-1.0.0_015
Tagging: 172.22.102.170:5000/abr2ts_release/fluent:cisco-1.0.0_015
Docker push: 172.22.102.170:5000/abr2ts_release/fluent:cisco-1.0.0_015
The push refers to a repository [172.22.102.170:5000/abr2ts_release/fluent]
946c30139423: Pushed
b33d84e048fb: Pushed
3eee1d1f03b4: Pushed
30b2eeafe84a: Pushed
80ff5081dbe2: Pushed
15172e48c360: Pushed
6b416303629f: Pushed
c21cd402f80f: Pushed
8019f73b25da: Pushed
2f924f30f505: Pushed
1352fbf1785a: Pushed
440537231155: Pushed
9ef0e71a9476: Pushed
dedb3c7497d1: Pushed
c3383b41c607: Pushed
088302c22591: Pushed
2aff46eaac4c: Pushed
98d85932a25b: Pushed
```

```
9f2bba04a565: Pushed
eb6427fb215b: Pushed
d66bf69bed8d: Pushed
5695a7ee01c0: Pushed
d938d6655758: Pushed
625d45015bed: Pushed
3dea68c34942: Pushed
efdf063314e7: Pushed
367b9c52c931: Pushed
3567b2f05514: Pushed
292a66992f77: Pushed
641fcd2417bc: Pushed
78ff13900d61: Pushed
cisco-1.0.0_015: digest: sha256:73072491061012ffc0e70790d3550717e34f45603d0bcacb0889591113e24123 size: 6791
The push refers to a repository [172.22.102.170:5000/abr2ts_release/fluent]
946c30139423: Layer already exists
b33d84e048fb: Layer already exists
3eee1d1f03b4: Layer already exists
30b2eeafe84a: Layer already exists
80ff5081dbe2: Layer already exists
15172e48c360: Layer already exists
6b416303629f: Layer already exists
c21cd402f80f: Layer already exists
8019f73b25da: Layer already exists
2f924f30f505: Layer already exists
1352fbf1785a: Layer already exists
440537231155: Layer already exists
9ef0e71a9476: Layer already exists
dedb3c7497d1: Layer already exists
c3383b41c607: Layer already exists
088302c22591: Layer already exists
2aff46eaac4c: Layer already exists
98d85932a25b: Layer already exists
9f2bba04a565: Layer already exists
eb6427fb215b: Layer already exists
d66bf69bed8d: Layer already exists
5695a7ee01c0: Layer already exists
d938d6655758: Layer already exists
625d45015bed: Layer already exists
3dea68c34942: Layer already exists
efdf063314e7: Layer already exists
367b9c52c931: Layer already exists
3567b2f05514: Layer already exists
292a66992f77: Layer already exists
641fcd2417bc: Layer already exists
78ff13900d61: Layer already exists
latest: digest: sha256:73072491061012ffc0e70790d3550717e34f45603d0bcacb0889591113e24123 size: 6791
Untagged: abr2ts_release/fluent:cisco-1.0.0_015
Untagged:
172.22.102.170:5000/abr2ts_release/fluent@sha256:73072491061012ffc0e70790d3550717e34f45603d0bcacb0889591113e2
4123
[root@cmt-deployer scripts]#
```

Step 4 Run the following command:

```
./load_to_registry_infra.sh {Deployer_Server_IP}
```

for example:

```
[root@cmt-deployer scripts]# ./load_to_registry_infra.sh 172.22.102.170
```

Output (excerpts from the beginning and end)

```
IPVS_TAG cisco-1.0.0-26.
1.0.0-26
```

```
docker load -i ../abr2ts-docker-images/images/ipvs_keepalived.cisco-1.0.0-26.tar
4f1bf1d2e24a: Loading layer [==================================================>] 5.632 kB/5.632 kB
a1653ba4e89e: Loading layer [==================================================>] 4.608 kB/4.608 kB
aec3772817ad: Loading layer [==================================================>] 45.46 MB/45.46 MB
6d0beb5e1a3d: Loading layer [==================================================>] 2.048 kB/2.048 kB
f2169e2b88db: Loading layer [==================================================>] 2.103 MB/2.103 MB
f6b7d9cefd18: Loading layer [==================================================>] 4.608 kB/4.608 kB
0ec33decef5c: Loading layer [==================================================>] 72.34 MB/72.34 MB
9620c16b12fe: Loading layer [==================================================>] 8.704 kB/8.704 kB
062f7ba27df8: Loading layer [==================================================>]   240 MB/240 MB
29f4e5bd990e: Loading layer [==================================================>] 2.084 MB/2.084 MB
240eaa2aed44: Loading layer [==================================================>] 2.048 kB/2.048 kB
Loaded image: dockerhub.cisco.com/spvss-vmp-docker-dev/vmp/cipvs/ipvs_keepalived:1.0.0-26
Prev Tag: dockerhub.cisco.com/spvss-vmp-docker-dev/vmp/cipvs/ipvs_keepalived:1.0.0-26
Tagging: 172.22.102.170:5000/cisco_ipvs_keepalived_os_release/ipvs_keepalived:cisco-1.0.0-26
Docker push: 172.22.102.170:5000/cisco_ipvs_keepalived_os_release/ipvs_keepalived:cisco-1.0.0-26
The push refers to a repository [172.22.102.170:5000/cisco_ipvs_keepalived_os_release/ipvs_keepalived]
240eaa2aed44: Pushed
29f4e5bd990e: Pushed
062f7ba27df8: Pushed
9620c16b12fe: Pushed
0ec33decef5c: Pushed
f6b7d9cefd18: Pushed
f2169e2b88db: Pushed
6d0beb5e1a3d: Pushed
aec3772817ad: Pushed
a1653ba4e89e: Pushed
4f1bf1d2e24a: Pushed
34e7b85d83e4: Mounted from openshift/origin-metrics-hawkular-metrics
cisco-1.0.0-26: digest: sha256:373d1ebbb74cf5e68f543a4bed3c05c0d9d2c77e0c1760eee5bd4c066e140799 size: 2828
The push refers to a repository [172.22.102.170:5000/cisco_ipvs_keepalived_os_release/ipvs_keepalived]
240eaa2aed44: Layer already exists
29f4e5bd990e: Layer already exists
062f7ba27df8: Layer already exists
9620c16b12fe: Layer already exists
0ec33decef5c: Layer already exists
f6b7d9cefd18: Layer already exists
f2169e2b88db: Layer already exists
6d0beb5e1a3d: Layer already exists
aec3772817ad: Layer already exists
a1653ba4e89e: Layer already exists
4f1bf1d2e24a: Layer already exists
34e7b85d83e4: Layer already exists
latest: digest: sha256:373d1ebbb74cf5e68f543a4bed3c05c0d9d2c77e0c1760eee5bd4c066e140799 size: 2828
Untagged: dockerhub.cisco.com/spvss-vmp-docker-dev/vmp/cipvs/ipvs_keepalived:1.0.0-26
docker load -i ../abr2ts-docker-images/images/prometheus.cisco-1.0.0_013.tar.gz
6a749002dd6a: Loading layer [==================================================>] 1.338 MB/1.338 MB
5f70bf18a086: Loading layer [==================================================>] 1.024 kB/1.024 kB
1692ded805c8: Loading layer [==================================================>] 2.629 MB/2.629 MB
f48243dac885: Loading layer [==================================================>] 60.85 MB/60.85 MB
cdd8264671af: Loading layer [==================================================>] 11.95 MB/11.95 MB
9e57a85f391a: Loading layer [==================================================>] 3.584 kB/3.584 kB
47386f4e4480: Loading layer [==================================================>] 15.36 kB/15.36 kB
25f242732872: Loading layer [==================================================>] 68.61 kB/68.61 kB
0c5a94346a1e: Loading layer [==================================================>] 10.75 kB/10.75 kB
1766652c0ba6: Loading layer [==================================================>] 1.536 kB/1.536 kB
Loaded image: abr2ts_release/prometheus:cisco-1.0.0_013
Prev Tag: abr2ts_release/prometheus:cisco-1.0.0_013
Tagging: 172.22.102.170:5000/abr2ts_release/prometheus:cisco-1.0.0_013
Docker push: 172.22.102.170:5000/abr2ts_release/prometheus:cisco-1.0.0_013
The push refers to a repository [172.22.102.170:5000/abr2ts_release/prometheus]
1766652c0ba6: Pushed
5f70bf18a086: Mounted from openshift/origin-logging-auth-proxy
0c5a94346a1e: Pushed
```

```
25f242732872: Pushed
47386f4e4480: Pushed
9e57a85f391a: Pushed
cdd8264671af: Pushed
f48243dac885: Pushed
1692ded805c8: Pushed
6a749002dd6a: Pushed
cisco-1.0.0_013: digest: sha256:6b4dae8aca870ada8624496ecf2715d0706dd0e1c3e439fcc680a766991d1c80 size: 3638
The push refers to a repository [172.22.102.170:5000/abr2ts_release/prometheus]
1766652c0ba6: Layer already exists
5f70bf18a086: Layer already exists
0c5a94346a1e: Layer already exists
25f242732872: Layer already exists
47386f4e4480: Layer already exists
9e57a85f391a: Layer already exists
cdd8264671af: Layer already exists
f48243dac885: Layer already exists
1692ded805c8: Layer already exists
6a749002dd6a: Layer already exists
latest: digest: sha256:6b4dae8aca870ada8624496ecf2715d0706dd0e1c3e439fcc680a766991d1c80 size: 3638
Untagged: abr2ts_release/prometheus:cisco-1.0.0_013
Untagged:
172.22.102.170:5000/abr2ts_release/prometheus@sha256:6b4dae8aca870ada8624496ecf2715d0706dd0e1c3e439fcc680a766
991d1c80
docker load -i ../abr2ts-docker-images/images/grafana.cisco-1.0.0_013.tar.gz
c01c63c6823d: Loading layer [==================================================>] 129.3 MB/129.3 MB
5f70bf18a086: Loading layer [==================================================>] 1.024 kB/1.024 kB
e09843e376c0: Loading layer [==================================================>] 163.6 MB/163.6 MB
e1d1aab8e861: Loading layer [==================================================>] 3.584 kB/3.584 kB
Loaded image: abr2ts_release/grafana:cisco-1.0.0_013
Prev Tag: abr2ts_release/grafana:cisco-1.0.0_013
Tagging: 172.22.102.170:5000/abr2ts_release/grafana:cisco-1.0.0_013
Docker push: 172.22.102.170:5000/abr2ts_release/grafana:cisco-1.0.0_013
The push refers to a repository [172.22.102.170:5000/abr2ts_release/grafana]
5f70bf18a086: Mounted from abr2ts_release/prometheus
e1d1aab8e861: Pushed
e09843e376c0: Pushed
c01c63c6823d: Pushed
cisco-1.0.0_013: digest: sha256:9994cd155cc35ff40c48b68914c485db388f7c70c442f3ac5f9ee3933c96115d size: 1772
The push refers to a repository [172.22.102.170:5000/abr2ts_release/grafana]
5f70bf18a086: Layer already exists
e1d1aab8e861: Layer already exists
e09843e376c0: Layer already exists
c01c63c6823d: Layer already exists
latest: digest: sha256:9994cd155cc35ff40c48b68914c485db388f7c70c442f3ac5f9ee3933c96115d size: 1772
Untagged: abr2ts_release/grafana:cisco-1.0.0_013
Untagged:
172.22.102.170:5000/abr2ts_release/grafana@sha256:9994cd155cc35ff40c48b68914c485db388f7c70c442f3ac5f9ee3933c9
6115d
docker load -i ../abr2ts-docker-images/images/alertmanager.cisco-1.0.0_014.tar.gz
0271b8eebde3: Loading layer [==================================================>] 1.338 MB/1.338 MB
68d1a8b41cc0: Loading layer [==================================================>] 2.586 MB/2.586 MB
5f70bf18a086: Loading layer [==================================================>] 1.024 kB/1.024 kB
f1018bf28474: Loading layer [==================================================>] 15.26 MB/15.26 MB
acf1a861c3c8: Loading layer [==================================================>] 6.144 kB/6.144 kB
da41d7fe4034: Loading layer [==================================================>] 1.536 kB/1.536 kB
Loaded image: abr2ts_release/alertmanager:cisco-1.0.0_014
Prev Tag: abr2ts_release/alertmanager:cisco-1.0.0_014
Tagging: 172.22.102.170:5000/abr2ts_release/alertmanager:cisco-1.0.0_014
Docker push: 172.22.102.170:5000/abr2ts_release/alertmanager:cisco-1.0.0_014
The push refers to a repository [172.22.102.170:5000/abr2ts_release/alertmanager]
da41d7fe4034: Pushed
5f70bf18a086: Mounted from abr2ts_release/grafana
acf1a861c3c8: Pushed
```

```
f1018bf28474: Pushed
68d1a8b41cc0: Pushed
0271b8eebde3: Pushed
cisco-1.0.0_014: digest: sha256:85cd7f29e69cf6a2b2377b51715f70c020365529f7778f237d4f29f0b675e5d7 size: 2599
The push refers to a repository [172.22.102.170:5000/abr2ts_release/alertmanager]
da41d7fe4034: Layer already exists
5f70bf18a086: Layer already exists
acf1a861c3c8: Layer already exists
f1018bf28474: Layer already exists
68d1a8b41cc0: Layer already exists
0271b8eebde3: Layer already exists
latest: digest: sha256:85cd7f29e69cf6a2b2377b51715f70c020365529f7778f237d4f29f0b675e5d7 size: 2599
Untagged: abr2ts_release/alertmanager:cisco-1.0.0_014
Untagged:
172.22.102.170:5000/abr2ts_release/alertmanager@sha256:85cd7f29e69cf6a2b2377b51715f70c020365529f7778f237d4f29
f0b675e5d7
processing
b1b065555b8a: Loading layer [==================================================>] 202.2 MB/202.2 MB
8f88b13c186a: Loading layer [==================================================>]   142 MB/142 MB
4a6dd33b17c3: Loading layer [==================================================>] 188.4 MB/188.4 MB
a556a6ebe628: Loading layer [==================================================>]  45.5 MB/45.5 MB
167444722077: Loading layer [==================================================>] 45.68 MB/45.68 MB
1c428d195cbe: Loading layer [==================================================>] 1.536 kB/1.536 kB
d6049b101281: Loading layer [==================================================>] 6.144 kB/6.144 kB
265b57c75b75: Loading layer [==================================================>]  5.12 kB/5.12 kB
Loaded image: dockerhub.cisco.com/spvss-ivp-ci-release-docker/infra/kafka:20180110164016-0.10.2
Tagging: 172.22.102.170:5000/abr2ts_release/infra/kafka:20180110164016-0.10.2
Docker push: 172.22.102.170:5000/abr2ts_release/infra/kafka:20180110164016-0.10.2
The push refers to a repository [172.22.102.170:5000/abr2ts_release/infra/kafka]
265b57c75b75: Pushed
d6049b101281: Pushed
1c428d195cbe: Pushed
167444722077: Pushed
a556a6ebe628: Pushed
4a6dd33b17c3: Pushed
8f88b13c186a: Pushed
b1b065555b8a: Pushed
20180110164016-0.10.2: digest: sha256:0940d4a397acc37850a405c34b4a9c35be5a3e5b80d299754933c7fcec8b103f size:
2000
Docker rmi: dockerhub.cisco.com/spvss-ivp-ci-release-docker/infra/kafka:20180110164016-0.10.2
Untagged: dockerhub.cisco.com/spvss-ivp-ci-release-docker/infra/kafka:20180110164016-0.10.2
processing kafka-20180110164016-0.10.2.tar.gz
3b1715f19f7b: Loading layer [==================================================>] 96.09 MB/96.09 MB
71007b2b2cc9: Loading layer [==================================================>]  12.3 MB/12.3 MB
0b930375e377: Loading layer [==================================================>] 1.536 kB/1.536 kB
aa50aad3be21: Loading layer [==================================================>] 3.584 kB/3.584 kB
d9ec4e4c62d4: Loading layer [==================================================>]  2.56 kB/2.56 kB
Loaded image: dockerhub.cisco.com/spvss-ivp-ci-release-docker/infra/kafka-exporter:20180108142414-0.3.0
Tagging: 172.22.102.170:5000/abr2ts_release/infra/kafka-exporter:20180108142414-0.3.0
Docker push: 172.22.102.170:5000/abr2ts_release/infra/kafka-exporter:20180108142414-0.3.0
The push refers to a repository [172.22.102.170:5000/abr2ts_release/infra/kafka-exporter]
d9ec4e4c62d4: Pushed
aa50aad3be21: Pushed
0b930375e377: Pushed
71007b2b2cc9: Pushed
3b1715f19f7b: Pushed
34e7b85d83e4: Mounted from cisco_ipvs_keepalived_os_release/ipvs_keepalived
20180108142414-0.3.0: digest: sha256:d8dbee7ad773a080a2e0e1344828bfec5aa09124c50d3e34aeb93860a5adb8a2 size:
1574
Docker rmi: dockerhub.cisco.com/spvss-ivp-ci-release-docker/infra/kafka-exporter:20180108142414-0.3.0
Untagged: dockerhub.cisco.com/spvss-ivp-ci-release-docker/infra/kafka-exporter:20180108142414-0.3.0
processing kafka-exporter-20180108142414-0.3.0.tar.gz
c07856edd69d: Loading layer [==================================================>]  2.56 kB/2.56 kB
d2046a352e0a: Loading layer [==================================================>] 307.8 MB/307.8 MB
```

```
2bcf920f81c7: Loading layer [==================================================>] 8.192 kB/8.192 kB
3207f9da5ebe: Loading layer [==================================================>] 4.096 kB/4.096 kB
9b351fbb42c7: Loading layer [==================================================>] 8.192 kB/8.192 kB
Loaded image: dockerhub.cisco.com/spvss-ivp-ci-release-docker/lmm/logstash:20180108142323-5.5.0
Tagging: 172.22.102.170:5000/abr2ts_release/lmm/logstash:20180108142323-5.5.0
Docker push: 172.22.102.170:5000/abr2ts_release/lmm/logstash:20180108142323-5.5.0
The push refers to a repository [172.22.102.170:5000/abr2ts_release/lmm/logstash]
9b351fbb42c7: Pushed
3207f9da5ebe: Pushed
2bcf920f81c7: Pushed
d2046a352e0a: Pushed
c07856edd69d: Pushed
4a6dd33b17c3: Mounted from abr2ts_release/infra/kafka
8f88b13c186a: Mounted from abr2ts_release/infra/kafka
b1b065555b8a: Mounted from abr2ts_release/infra/kafka
20180108142323-5.5.0: digest: sha256:5843225a0c97edaf4f36df5b861554f198762883c4e50fd26a599b2538229c06 size:
1997
Docker rmi: dockerhub.cisco.com/spvss-ivp-ci-release-docker/lmm/logstash:20180108142323-5.5.0
Untagged: dockerhub.cisco.com/spvss-ivp-ci-release-docker/lmm/logstash:20180108142323-5.5.0
processing logstash-20180108142323-5.5.0.tar.gz
7a95a7bd92d3: Loading layer [==================================================>]  3.95 MB/3.95 MB
74bd5f06ca5e: Loading layer [==================================================>] 6.296 MB/6.296 MB
039203a306da: Loading layer [==================================================>] 2.048 kB/2.048 kB
a30c6f038542: Loading layer [==================================================>] 2.048 kB/2.048 kB
Loaded image: dockerhub.cisco.com/spvss-ivp-ci-release-docker/infra/proxytoservice:20180108143454-1.0.0
Tagging: 172.22.102.170:5000/abr2ts_release/infra/proxytoservice:20180108143454-1.0.0
Docker push: 172.22.102.170:5000/abr2ts_release/infra/proxytoservice:20180108143454-1.0.0
The push refers to a repository [172.22.102.170:5000/abr2ts_release/infra/proxytoservice]
a30c6f038542: Pushed
039203a306da: Pushed
74bd5f06ca5e: Pushed
7a95a7bd92d3: Pushed
7e3694659f4b: Mounted from abr2ts_release/vod-gateway
20180108143454-1.0.0: digest: sha256:bd4804358b290eefe85a7e2ec9e1dd0a0dc69aa3d381a4c476f500711d264dee size:
1364
Docker rmi: dockerhub.cisco.com/spvss-ivp-ci-release-docker/infra/proxytoservice:20180108143454-1.0.0
Untagged: dockerhub.cisco.com/spvss-ivp-ci-release-docker/infra/proxytoservice:20180108143454-1.0.0
processing proxytoservice-20180108143454-1.0.0.tar.gz
7cdc39d04de1: Loading layer [==================================================>]   137 MB/137 MB
05ea469fa0f1: Loading layer [==================================================>] 3.584 kB/3.584 kB
5bae38d9655f: Loading layer [==================================================>] 5.632 kB/5.632 kB
0b81297d1cdd: Loading layer [==================================================>] 2.991 MB/2.991 MB
3a8b8a86490e: Loading layer [==================================================>] 5.632 kB/5.632 kB
Loaded image: dockerhub.cisco.com/spvss-ivp-ci-release-docker/infra/zookeeper:20180108141946-3.5.2
Tagging: 172.22.102.170:5000/abr2ts_release/infra/zookeeper:20180108141946-3.5.2
Docker push: 172.22.102.170:5000/abr2ts_release/infra/zookeeper:20180108141946-3.5.2
The push refers to a repository [172.22.102.170:5000/abr2ts_release/infra/zookeeper]
3a8b8a86490e: Pushed
0b81297d1cdd: Pushed
5bae38d9655f: Pushed
05ea469fa0f1: Pushed
7cdc39d04de1: Pushed
4a6dd33b17c3: Mounted from abr2ts_release/infra/kafka
8f88b13c186a: Mounted from abr2ts_release/infra/kafka
b1b065555b8a: Mounted from abr2ts_release/infra/kafka
20180108141946-3.5.2: digest: sha256:1905c66ce6598ad932c1b10f7f9cbde4d1142d5dcbf2d8e7b4135ef919c3f6f8 size:
1997
Docker rmi: dockerhub.cisco.com/spvss-ivp-ci-release-docker/infra/zookeeper:20180108141946-3.5.2
Untagged: dockerhub.cisco.com/spvss-ivp-ci-release-docker/infra/zookeeper:20180108141946-3.5.2
processing zookeeper-20180108141946-3.5.2.tar.gz
13cb9e79b602: Loading layer [==================================================>]  3.95 MB/3.95 MB
088bb813236a: Loading layer [==================================================>] 130.1 MB/130.1 MB
5c6b3c8bd7e7: Loading layer [==================================================>] 2.975 MB/2.975 MB
f57337814357: Loading layer [==================================================>] 1.766 MB/1.766 MB
```

```
149408a43273: Loading layer [==================================================>] 7.396 MB/7.396 MB
e52a7754d659: Loading layer [==================================================>] 18.33 MB/18.33 MB
bcb379e41c6a: Loading layer [==================================================>] 20.93 MB/20.93 MB
80fb0142bfa0: Loading layer [==================================================>] 92.49 MB/92.49 MB
6700f57265ce: Loading layer [==================================================>] 10.17 MB/10.17 MB
a22b1f589fd4: Loading layer [==================================================>] 98.01 MB/98.01 MB
170f0bdedfd0: Loading layer [==================================================>] 172.5 kB/172.5 kB
cb330fba5027: Loading layer [==================================================>] 20.48 kB/20.48 kB
b7e8643a5645: Loading layer [==================================================>] 3.574 MB/3.574 MB
Loaded image: dockerhub.cisco.com/spvss-ivp-ci-release-docker/lmm/logging-bundle:20180110164043-17.4.3
Tagging: 172.22.102.170:5000/abr2ts_release/lmm/logging-bundle:20180110164043-17.4.3
Docker push: 172.22.102.170:5000/abr2ts_release/lmm/logging-bundle:20180110164043-17.4.3
The push refers to a repository [172.22.102.170:5000/abr2ts_release/lmm/logging-bundle]
b7e8643a5645: Pushed
cb330fba5027: Pushed
170f0bdedfd0: Pushed
a22b1f589fd4: Pushed
6700f57265ce: Pushed
80fb0142bfa0: Pushed
bcb379e41c6a: Pushed
e52a7754d659: Pushed
149408a43273: Pushed
f57337814357: Pushed
5c6b3c8bd7e7: Pushed
088bb813236a: Pushed
13cb9e79b602: Pushed
7e3694659f4b: Mounted from abr2ts_release/infra/proxytoservice
20180110164043-17.4.3: digest: sha256:c844b56cbd03ff10753f58437b6a6cdfbd81ba74eba772b4984400c8c68ebe7b size:
3271
Docker rmi: dockerhub.cisco.com/spvss-ivp-ci-release-docker/lmm/logging-bundle:20180110164043-17.4.3
Untagged: dockerhub.cisco.com/spvss-ivp-ci-release-docker/lmm/logging-bundle:20180110164043-17.4.3
[root@cmt-deployer bundle]#
```

Verifying Docker Image Loading

After the Docker image loading process has completed, you must verify that the images have been successfully tagged to the Docker registry.

Command

```
[root@cmt-deployer scripts]# docker images
```

Output

```
REPOSITORY                                                    TAG                   IMAGE ID           CREATED
SIZE
172.22.102.170:5000/abr2ts_release/fluent                     cisco-1.0.0_015       d922b97e10c7       5 days ago
780.7 MB
172.22.102.170:5000/abr2ts_release/fluent                     latest                d922b97e10c7       5 days ago
780.7 MB
172.22.102.170:5000/abr2ts_release/vod-gateway                cisco-1.0.0_3         70ed820ac341       5 days ago
12 MB
172.22.102.170:5000/abr2ts_release/vod-gateway                latest                70ed820ac341       5 days ago
12 MB
172.22.102.170:5000/abr2ts_release/alertmanager               cisco-1.0.0_014       abfa3119c673       11 days
ago         17.8 MB
172.22.102.170:5000/abr2ts_release/alertmanager               latest                abfa3119c673       11 days
ago         17.8 MB
172.22.102.170:5000/abr2ts_release/lmm/logging-bundle         20180110164043-17.4.3 495ca213fe5d       2 weeks
ago         384.1 MB
172.22.102.170:5000/abr2ts_release/infra/kafka                20180110164016-0.10.2 635cc17d2245       2 weeks
ago         610.7 MB
172.22.102.170:5000/abr2ts_release/infra/proxytoservice       20180108143454-1.0.0  92a01d33afde       3 weeks
ago         12.3 MB
172.22.102.170:5000/abr2ts_release/infra/kafka-exporter       20180108142414-0.3.0  ad5adf6a3d46       3 weeks
ago         297 MB
```

```
172.22.102.170:5000/abr2ts_release/lmm/logstash                          20180108142323-5.5.0    f206a935952f        3 weeks
ago         817.5 MB
172.22.102.170:5000/abr2ts_release/infra/zookeeper                       20180108141946-3.5.2    1910159f2b55        3 weeks
ago         657.1 MB
172.22.102.170:5000/cisco_ipvs_keepalived_os_release/ipvs_keepalived     cisco-1.0.0-26          eb9933eaf9c5        3 weeks
ago         547.7 MB
172.22.102.170:5000/cisco_ipvs_keepalived_os_release/ipvs_keepalived     latest                  eb9933eaf9c5        3 weeks
ago         547.7 MB
172.22.102.170:5000/abr2ts_release/prometheus                            cisco-1.0.0_013         237cd52aef24        7 weeks
ago         75.42 MB
172.22.102.170:5000/abr2ts_release/prometheus                            latest                  237cd52aef24        7 weeks
ago         75.42 MB
172.22.102.170:5000/abr2ts_release/grafana                               cisco-1.0.0_013         86f1955c7430        3 months
ago        285.2 MB
172.22.102.170:5000/abr2ts_release/grafana                               latest                  86f1955c7430        3 months
ago        285.2 MB
```

Create the ABR2TS Project Namespace

Within OpenShift, you can create unique namespaces that allow multiple projects to be managed simultaneously by Kubernetes. Pods are then started and stopped, under their own project namespace. For all intents and purposes, projects and namespaces can be considered the same.

To create the ABR2TS project namespace within OpenShift:

Step 1 Make sure that you are logged into the deployer node.

```
ssh root@<Deployer_Node_IP>
```

Step 2 Log into OpenShift.

```
oc login -u system -p admin --insecure-skip-tls-verify=true -n default
https://cmt-osp-cluster.cmtlab-dns.com:8443
```

Output

```
Login successful.

You have access to the following projects and can switch between them with 'oc project
<projectname>':

  * default
    kube-system
    logging
    management-infra
    openshift
    openshift-infra

Using project "default".
```

Step 3 Execute the following command to create the namepace:

```
kubectl create namespace abr2ts
```

Step 4 Execute the following command to verify that the “abr2ts” namespace has been created:

```
[root@cmt-deployer scripts]# kubectl get namespaces
```

Output

```
NAME              STATUS    AGE
abr2ts            Active    24s
default           Active    13h
```

```
kube-system        Active    13h
logging            Active    13h
management-infra   Active    13h
openshift          Active    13h
openshift-infra    Active    13h
```

Configuring VoD Gateway & Fluentd Pods

Next, you will need to configure the CMT and Fluentd pods by using the following procedures.

Step 1 Change to the abr2ts project:

```
[root@platform cmt-deployment]# oc project abr2ts
```

Output

```
Now using project "abr2ts" on server "https://cmt-osp-cluster.cmtlab-dns.com:8443”.
```

Step 2 Get the ABR2TS context view.

```
[root@platform scripts]# kubectl config view
```

Output

```
apiVersion: v1
clusters:
- cluster:
    insecure-skip-tls-verify: true
    server: https://172.22.102.244:8443
  name: 172-22-102-244:8443
- cluster:
    api-version: v1
    insecure-skip-tls-verify: true
    server: https://cmt-osp-cluster.cmtlab-dns.com:8443
  name: cmt-osp-cluster-cmtlab-dns-com:8443
contexts:
- context:
    cluster: cmt-osp-cluster-cmtlab-dns-com:8443
    namespace: abr2ts
    user: system/cmt-osp-cluster-cmtlab-dns-com:8443
  name: abr2ts/cmt-osp-cluster-cmtlab-dns-com:8443/system
- context:
    cluster: 172-22-102-244:8443
    namespace: default
    user: system/172-22-102-244:8443
  name: default/172-22-102-244:8443/system
- context:
    cluster: cmt-osp-cluster-cmtlab-dns-com:8443
    namespace: default
    user: system/cmt-osp-cluster-cmtlab-dns-com:8443
  name: default/cmt-osp-cluster-cmtlab-dns-com:8443/system
current-context: abr2ts/cmt-osp-cluster-cmtlab-dns-com:8443/system
kind: Config
preferences: {}
users:
- name: system/172-22-102-244:8443
  user:
    token: 1_HbRDy8n1W-E-94T823fG8C6o-Z5jCUr5RbuufA0Wg
- name: system/cmt-osp-cluster-cmtlab-dns-com:8443
  user:
    token: JqBkXklj9j1DnjV0gfdkHCA014bwbbc4DYodY6dP9yI
```

Step 3 Edit the file below and update the bold values.

`/root/abr2ts-deployment/abr2ts.cfg file`

File Contents

```
{
        "siteId": "ciscok8s",
        "abr2tsRootPath": "/root/abr2ts-deployment",
```

Note The abr2tsContext field value should be updated to match the `current-context` value in step 2.

```
        "abr2tsContext": "abr2ts/cmt-osp-cluster-cmtlab-dns-com:8443/system",
        "dockerRegIp": "172.22.102.170", #deployer IP address
        "k8sMaster": "172.22.102.244", #LB VIP IP address
        "abr2tsServiceIp": "172.22.97.44",
        "abr2tsServiceFqdn": "172.22.97.44",
        "openshiftPlatform": "Yes",
        "httpsProxy": "",
        "httpProxy": "",
        "kafkaDefaultTopic": "logs",
        "kafkaIhPort": "127.0.0.1:2182",
        "usernameToken": "",
        "abr2tsSelfHostname": "abr2ts-oc.cisco.com",
        "logServer": "172.22.98.70",
        "logServerType": "COLLECTOR",
        "logCollectorTcpAddr": "lmm-logstash-logcollector.infra.svc.cluster.local",
        "platformInfo": {
                "path": "platform/resources/config",
                "packageName": "cisco-k8s-upic",
        }
}
```

Step 4 Change directory to `/root/abr2ts-deployment/scripts`.

Step 5 Run the following command to configure the CMT and Fluentd pods:

```
./abr2ts_vod_gateway.sh  config
```

Step 6 Navigate to the following directory:

```
cd /root/abr2ts-deployment/platform/resources/config/vod-gateway
```

Step 7 Edit the file: `vod-gateway-rc.json` to update the "replicas": 20," field to reflect the desired number of CMT worker pods in your cluster.

Note Each worker node can run a maximum of 5 pods.

Logging Queue Deployment

The Logging Queue consists of a number of services that provide functionality to export system metrics to external systems or components, such as Splunk or Elk.

This section will first describe the components that make up the logging queue. Next, it will explain the procedures for deploying the logging queue onto your cluster.

The logging queue consists of the following components:

- Logstash - consists of two components: Log collector and Log pusher. Log collector receives logs from Fluentd and forwards them to Kafka. Log pusher, pushes logs to a specific destination (via TCP) such as Elastic Search or Splunk.
- Kafka & Kafka Exporter - each Kafka broker runs as a pod and service set. To be accessed from outside the cluster, Kafka uses hostport; where one broker runs on one host.
- Zookeeper - is a component that tracks the status of the cluster for Kafka.
- logging-bundle-ansible-bundle
- Proxy-to-service - is a proxy that port-forwards requests to the appropriate service.

The logging deployment scripts are located at:
`/root/abr2ts-deployment/logging-bundle-20180110164043-17.4.3`

Configuring the Logging Queue

To configure the logging queue, perform the following steps:

Step 1 Navigate to:
`/root/abr2ts-deployment/logging-bundle-20180110164043-17.4.3/bundle/inventories`

Step 2 Update `abr2ts.ini` to update the values shown in bold below:

File Contents

```
localhost   ansible_connection=local

[all]
[all:vars]
docker_registry_path=172.22.102.170:5000/abr2ts_release # <Deployer IP>

namespace=infra

kubernetes_flavor=openshift
openshift_master=cmt-osp-cluster.cmtlab-dns.com:8443 # <Load Balancer VIP Hostname>
openshift_user=system
openshift_password=admin
logging_queue_enable_logpusher=true
logging_queue_logpusher_tcp_host=172.22.102.57 # {Splunk Server IP}
logging_queue_logpusher_tcp_port=9995 # {Splunk Server Port}
logstash_output_tcp_codec=json
logging_queue_logpusher_target=tcp

enable_log_queue=true
logging_queue_enable_logcollectorproxy=true
#logging_queue_enable_logpusher=True
kafka_node_selectors="kubernetes.io/hostname: cmt-infra1, kubernetes.io/hostname:
cmt-infra2, kubernetes.io/hostname: cmt-infra3" # These are Infra Node Hostnames
zookeeper_node_selector="infra.cisco.com/type: infra" # Infra Node Labels
proxy_to_service_node_selectors="kubernetes.io/hostname: cmt-infra2"# Infra Hostname
logstash_node_selector="infra.cisco.com/type: infra" # Infra Node Labels

enable_log_visualisation=false

## In default mode - master, data and client/ingest Elastic nodes are seperate
# elasticsearch_master_replicas=3
# elasticsearch_data_replicas=3
```

```
# elasticsearch_client_replicas=2

## To combine Elastic node functions use the elasticsearch_mode=combine option
# elasticsearch_mode=combined
# elasticsearch_replicas=3

# elasticsearch_minimum_master_node=2

# dns_domain=infra.mydomain.com
[root@cmt-deployer ~]#
```

Step 3 Navigate to the logging queue deployment folder at:

```
/root/abr2ts-deployment/logging-bundle-20180110164043-17.4.3/logstash_pusher/logstash
```

Step 4 Edit the `deploy.yml` file to update the values shown in bold:

File Contents

```
#
# Copyright (c) 2017 Cisco Systems Inc., All rights reserved.
#
---

 - name: Deploy Logstash Pusher
   hosts: localhost
   connection: local
   gather_facts: no
   roles:
    - {
        role: config-openshift,
        when: kubernetes_flavor == "openshift"
      }
    - { role: kube-namespace }
    - { role: logstash, logstash_deployment_tag: "logpusher-splunk",
                        logstash_replicas: 1,
                        logstash_inputs: "kafka",
                        logstash_outputs: "tcp",
                        logstash_output_tcp_host: "172.22.102.57",#Splunk IP
                        logstash_output_tcp_port: 9995,
                        logstash_output_tcp_codec: "json",
                        logstash_input_kafka_bootstrap_servers:
"infra-kafka-0:9092,infra-kafka-1:9092,infra-kafka-2:9092",
                        logstash_input_kafka_topics: "ivp",
                        logstash_input_kafka_codec: "json"
    }
```

Deploying the Logging Queue to the Cluster

Next, you will use the following process to deploy the Logging Queue to the cluster.

Step 1 Navigate to /`root/abr2ts-deployment/logging-bundle-20180110164043-17.4.3/bundle`

Step 2 Run the script that will deploy the Logging Queue bundle.

```
# ./infra_logging.sh <deployer IP> start
```

Output (truncated)

```
Deploying Logging bundle
Running Ansible playbook
```

```
PLAY [Deploy Logging Bundle]
*****************************************************************************************
*************************************************

TASK [config-openshift : debug]
*****************************************************************************************
*********************************************
ok: [localhost] => {
    "msg": "Setting up OpenShift client for master cmt-osp-cluster.cmtlab-dns.com:8443"
}

TASK [config-openshift : Login to OpenShift master]
*****************************************************************************************
****************************
changed: [localhost]

TASK [kube-deploy : debug]
*****************************************************************************************
************************************************
ok: [localhost] => {
    "msg": "Processing templates for kube-namespace role"
}

...
... #Output has been truncated.
...

TASK [kube-deploy : Lookup the generated K8 resource type files]
*****************************************************************************************
****************
ok: [localhost] =>
(item=/root/abr2ts-deployment/logging-bundle-20180110164043-17.4.3/logstash_pusher/logstas
h/roles/logstash/generated/logstash-svc.yml)
ok: [localhost] =>
(item=/root/abr2ts-deployment/logging-bundle-20180110164043-17.4.3/logstash_pusher/logstas
h/roles/logstash/generated/logstash-deploy.yml)
ok: [localhost] =>
(item=/root/abr2ts-deployment/logging-bundle-20180110164043-17.4.3/logstash_pusher/logstas
h/roles/logstash/generated/logstash-config.yml)

TASK [kube-deploy : Apply templates]
*****************************************************************************************
*****************************************
changed: [localhost] =>
(item=/root/abr2ts-deployment/logging-bundle-20180110164043-17.4.3/logstash_pusher/logstas
h/roles/logstash/generated/logstash-svc.yml)
changed: [localhost] =>
(item=/root/abr2ts-deployment/logging-bundle-20180110164043-17.4.3/logstash_pusher/logstas
h/roles/logstash/generated/logstash-deploy.yml)
changed: [localhost] =>
(item=/root/abr2ts-deployment/logging-bundle-20180110164043-17.4.3/logstash_pusher/logstas
h/roles/logstash/generated/logstash-config.yml)

PLAY RECAP
*****************************************************************************************
*******************
localhost : ok=19 changed=7 unreachable=0 failed=0

Wait for 30 sec or so to complete the deployment
Now using project "infra" on server "https://cmt-osp-cluster.cmtlab-dns.com:8443".
pod "infra-kafka-0-1260527882-7f4wj" deleted
pod "infra-kafka-1-1923227918-9l41r" deleted
pod "infra-kafka-2-2585927954-nmgng" deleted
******************
```

```
Deploying Logging bundle Complete
[root@cmt-deployer bundle]#
```

Step 3 To verify the deployed state, you can use the following commands. The first command switches focus onto the infra project, while the second command displays the detailed status of the running pods.

Command 1 of 2:

```
[root@cmt-deployer bundle]# oc project infra
```

Output

```
Now using project "infra" on server "https://cmt-osp-cluster.cmtlab-dns.com:8443".
```

Command 2 of 2:

```
[root@cmt-deployer scripts]# oc get all
```

Output

```
NAME                                    DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
deploy/infra-kafka-0                    1         1         1            1           1h
deploy/infra-kafka-1                    1         1         1            1           1h
deploy/infra-kafka-2                    1         1         1            1           1h
deploy/infra-proxytoservice             1         1         1            1           1h
deploy/lmm-logstash-logcollector        2         2         2            2           1h
deploy/lmm-logstash-logpusher-splunk    1         1         1            1           1h

NAME                                CLUSTER-IP       EXTERNAL-IP   PORT(S)
AGE
svc/infra-kafka-0                   172.30.99.170    <none>        9092/TCP,9308/TCP
1h
svc/infra-kafka-1                   172.30.51.227    <none>        9092/TCP,9308/TCP
1h
svc/infra-kafka-2                   172.30.250.211   <none>        9092/TCP,9308/TCP
1h
svc/infra-zookeeper                    None             <none>
2888/TCP,3888/TCP,2181/TCP   1h
svc/lmm-logstash-logcollector       172.30.45.114    <none>        5000/TCP,4000/TCP
1h
svc/lmm-logstash-logpusher-splunk   172.30.70.210    <none>        5000/TCP,4000/TCP
1h

NAME                              DESIRED   CURRENT   AGE
statefulsets/infra-zookeeper      3         3         1h

NAME                                          DESIRED   CURRENT   READY     AGE
rs/infra-kafka-0-1260527882                   1         1         1         1h
rs/infra-kafka-1-1923227918                   1         1         1         1h
rs/infra-kafka-2-2585927954                   1         1         1         1h
rs/infra-proxytoservice-2125679694            1         1         1         1h
rs/lmm-logstash-logcollector-2105705753       2         2         2         1h
rs/lmm-logstash-logpusher-splunk-3483863117   1         1         1         1h

NAME                                             READY     STATUS    RESTARTS   AGE
po/infra-kafka-0-1260527882-tnbrx                2/2       Running   2          10m
po/infra-kafka-1-1923227918-4xdrr                2/2       Running   0          1h
po/infra-kafka-2-2585927954-vrvr1                2/2       Running   2          31m
po/infra-proxytoservice-2125679694-dfcss         1/1       Running   0          1h
po/infra-zookeeper-0                             1/1       Running   0          1h
po/infra-zookeeper-1                             1/1       Running   0          1h
po/infra-zookeeper-2                             1/1       Running   1          1h
po/lmm-logstash-logcollector-2105705753-1x8lg    1/1       Running   0          1h
po/lmm-logstash-logcollector-2105705753-n7jc1    1/1       Running   0          1h
```

```
po/lmm-logstash-logpusher-splunk-3483863117-hwb4c   1/1       Running   0          1h
[root@cmt-deployer scripts]#
```

Step 4 In order to stop the logging bundle, use the following command:

```
# ./infra_logging.sh <deployer IP> stop
```

Output

```
[root@ivpcoe-deployer bundle]# ./infra_logging.sh 172.22.102.170 stop
Destroying Logging bundle
Running Ansible playbook

PLAY [Delete Logging Bundle]
*********************************************************************************************
*********************************************************************************************
*********************

TASK [config-openshift : debug]
*********************************************************************************************
*********************************************************************************************
******************
ok: [localhost] => {
    "msg": "Setting up OpenShift client for master 172.22.98.80:8443"
}

TASK [config-openshift : Login to OpenShift master]
*********************************************************************************************
*********************************************************************************
changed: [localhost]

TASK [debug]
*********************************************************************************************
*********************************************************************************************
************************************
ok: [localhost] => {
    "msg": "About to destroy namespace infra"
}

TASK [pause]
*********************************************************************************************
*********************************************************************************************
************************************
```

Note Pay special attention to the steps below that are required to stop the pods.

```
[pause]
Press return to continue, Ctrl+C then "a" to abort:
ok: [localhost]

TASK [Delete Deployments]
*********************************************************************************************
*********************************************************************************************
************************
changed: [localhost]

TASK [Delete Stateful Sets]
*********************************************************************************************
*********************************************************************************************
**********************
changed: [localhost]
```

```
TASK [Delete ReplicaSets]
*********************************************************************************************
*********************************************************************************************
************************
changed: [localhost]

TASK [Delete Pods]
*********************************************************************************************
*********************************************************************************************
*****************************
changed: [localhost]

TASK [Delete Config Maps]
*********************************************************************************************
*********************************************************************************************
************************
changed: [localhost]

TASK [Delete Services]
*********************************************************************************************
*********************************************************************************************
*************************
changed: [localhost]

TASK [Delete Routes]
*********************************************************************************************
*********************************************************************************************
*****************************
changed: [localhost]

PLAY RECAP
*********************************************************************************************
*********************************************************************************************
***************************************
localhost                  : ok=11   changed=8    unreachable=0    failed=0

[root@ivpcoe-deployer bundle]#
```

Starting VoD Gateway & Fluentd

Running a standalone script will bring up the CMT and Fluentd logging services. Additionally, the script will deploy all required pods and verify that they are properly running.

Step 1 If necessary, SSH into the Deployer node as root.

Step 2 Change to the scripts folders.

```
cd /root/abr2ts-deployment/scripts
```

Step 3 Execute the following command:

```
[root@cmt-deployer scripts]# ./abr2ts_vod_gateway.sh start
```

Output

```
Kubernetes master is running at https://cmt-osp-cluster.cmtlab-dns.com:8443

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
Set context
Switched to context "abr2ts/cmt-osp-cluster-cmtlab-dns-com:8443/system".

Starting abr2ts vod_gateway
```

```
Starting ABR2TS K8S services
Context "abr2ts/cmt-osp-cluster-cmtlab-dns-com:8443/system" set.

2018-02-01 04:27:31 Starting vod-gateway service
service "vod-gateway" created
2018-02-01 04:27:31 vod-gateway service started successfully

Starting ABR2TS K8S pods
Context "abr2ts/cmt-osp-cluster-cmtlab-dns-com:8443/system" set.
2018-02-01 04:27:32 checking pods. kubeconfig=/root/.kube/config
2018-02-01 04:27:32 pic instance = abr2ts/cmt-osp-cluster-cmtlab-dns-com:8443/system
Context "abr2ts/cmt-osp-cluster-cmtlab-dns-com:8443/system" set.
2018-02-01 04:27:42 Checking if all nodes are in ready state
2018-02-01 04:27:42 all 9 nodes are in ready state

2018-02-01 04:27:42 starting vod-gateway rc
replicationcontroller "vod-gateway" created
2018-02-01 04:27:43 vod-gateway rc started

2018-02-01 04:27:43 Starting fluent
daemonset "fluent" created
2018-02-01 04:27:43 fluentd started

[root@cmt-deployer scripts]#
```

Verifying VoD Gateway & Fluentd Startup

To verify that all services, pods, and routes are up and running, execute the following commands:

```
# oc project abr2ts
```

First, make sure that you have switched into the CMT project if you are not already there.

```
# oc get pods -o wide
```

Provides a listing of all of the pods running in the cluster. At this stage, only Fluentd and CMT pods will be running. The Fluentd pods should be running on each node. The “o-wide” option shows you on which node each pod is running and the given IP address for each pod.

Note If you observe any issues while starting CMT or the Fluentd services, you should stop the services with “stop mode” prior to attempting to restart them. For details, see Stopping VoD Gateway & Fluentd, page 3-43.

Stopping VoD Gateway & Fluentd

The procedures in this section will stop the CMT service, the Fluentd daemon, and remove all related pods from the cluster.

To stop the CMT and Fluentd pods, run the CMT script in stop mode:

Step 1 Run the following command:

```
[root@cmt-deployer]# ./abr2ts_vod_gateway.sh stop
```

Step 2 Verify that all pods, the CMT service, and Fluentd are deleted. The following command verifies that all pods for the given namespace have been stopped.

```
[root@cmt-deployer]# oc get pods --namespace=abr2ts
```

Output:

```
No resources found.
```

Step 3 This command verifies that all services for the given namespace have been stopped.

```
[root@cmt-deployer]# oc get svc --namespace=abr2ts
```

Output:

```
No resources found
```

Configuring Splunk for use with CMT

The following steps are used to configure Splunk so that it can receive logs from CMT.

Note Cisco has tested CMT using Splunk Enterprise 6.6.

Step 1 Configure the Splunk server to accept TCP messages on port 9995. This configuration enables a log pusher that sends log events to Splunk over TCP using a JSON codec.

Step 2 The logging_queue_logpusher_tcp_port has been set to match the port specified in the `inputs.conf` file (located at `opt/splunk/etc/apps/abr2ts-splunk-config/local/`) in Splunk. The file contents are shown below:

File Contents

```
################
# Accept data from any host over TCP port 9995
#############
[tcp://:9995]
connection_host = dns
sourcetype = vod-gateway
source = tcp:9995
```

Step 3 A **props.conf** file (located at `opt/splunk/etc/apps/abr2ts-splunk-config/local/`) is used to properly split lines and to use the timestamp associated with the original log message. The file contents are shown below:

File Contents

```
[vod-gateway]
# from abr2ts
#
{"@timestamp":"2018-01-18T00:44:06.000Z","log":{"timeStamp":"2018-01-18T00:44:06.992Z","co
mponent":"abr2ts-vg","rxBytes":466
#
{"@timestamp":"2018-01-09T20:59:13.000Z","log":{"timeStamp":"2018-01-09T20:59:13.703Z","co
mponent":"abr2ts-vg","level":"INFO","module":"httpServer","FCID":"5d5b2eaa-1848-40aa-ad51-
b380642f8ad4","api":"/keepalive","httpMethod":"GET","url":"localhost/keepalive"},"stream":
"stdout","port":55570,"@version":"1","host":"10.129.0.1","time":"2018-01-09T20:59:13.70453
5022Z","container_id":"vod-gateway-mc65h_abr2ts_vod-gateway-220f22b92a59eb4c6b575edac95daa
69eb045e4db32c726da5c595ccca462011","tags":["logs.kubernetes.var.log.containers.vod-gatewa
y-mc65h_abr2ts_vod-gateway-220f22b92a59eb4c6b575edac95daa69eb045e4db32c726da5c595ccca46201
1.log"]}
TRUNCATE=0
SHOULD_LINEMERGE=false
# this one matches RLG logs parsed with codec=>json in the file{} filter
```

```
LINE_BREAKER=(\s*)\{"@timestamp
MAX_TIMESTAMP_LOOKAHEAD=50
# from regexr.com
TIME_PREFIX={\\"timeStamp\\":\\"
TIME_FORMAT=%Y-%m-%dT%H:%M:%S,%3N
KV_MODE=json
```

Verifying Connectivity with Splunk

The following procedures will verify that the Media Transformer and Splunk logging systems are communicating with each other correctly.

Step 1 First, we will need to verify that the Media Transformer logs are being received correctly. Start by logging into the Splunk user interface.

Step 2 Navigate to **App > Search**.

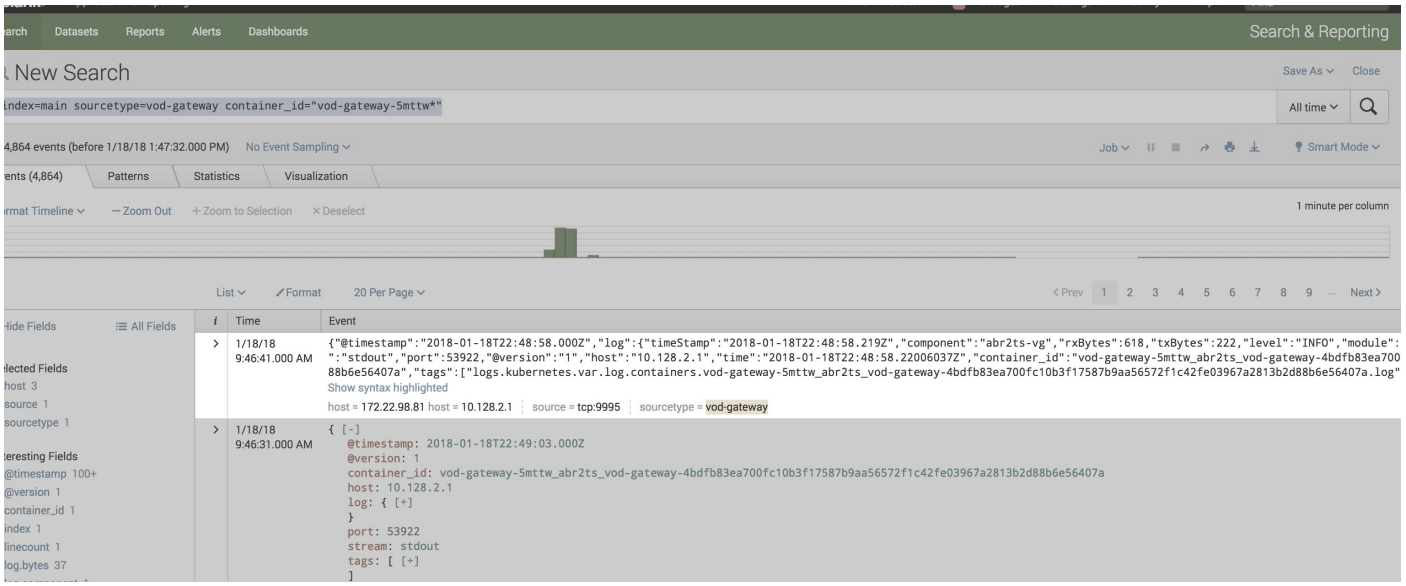
Step 3 Click on **Data Summary**.

Step 4 To retrieve Media Transformer log data, you will need to add a new search. Copy the following query into the Search field (near the top of the UI) and click the magnifying glass icon to execute the query:

```
index=main sourcetype=vod-gateway container_id="vod-gateway*"
```

Step 5 Verify that the Media Transformer logs have been successfully retrieved by choosing the “All time” date/time range. Log event records should appear in the interface.

Figure 3-2 Splunk “All time” option for log records



Configuring IPVS

Next, you must set values within an IPVS configuration file named `ipvs_service_configure.json`.

To update the IPVS configuration file:

Step 1 If necessary, SSH into the deployer node.

Step 2 Navigate to the scripts directory:

```
cd /root/abr2ts-deployment/cisco-ipvs-os/deployment/scripts
```

Step 3 Edit the file `ipvs_service_configure.json` to update the values that are shown in bold below.

```
{
  "name": "ipvs service conifg for keepalived",
  "version": "1.0.0",
  "deployment-config": {
    "namespace": "ipvs-service",
    "pod-resource": {
      "CPU": "1",
      "Memory": "1Gi"
    },
    "node-selector": {
      "ipvs-director-key": "cisco.com/type", //Node label as per inventory file (master)
      "ipvs-director-value": "master",
      "ipvs-backend-key": "cisco.com/type",
      "ipvs-backend-value": "backend" //Node label as per inventory file (backend)
    },
    "image": {
      "registry": "172.22.102.170:5000/cisco_ipvs_keepalived_os_release/ipvs_keepalived",
//Docker registry IP (Deployer IP)
      "image-version": "latest"
    }
  },
  "ipvs-config": {
    "vip": "192.169.131.1", //IPVS VIP
    "port": "80",
    "network_mask": "255.255.255.0", //IPVS VIP Netmask
    "service-ns": "abr2ts", //Namespace
    "service-identifier": "vod-gateway", //Service name
    "active-director-ip": "192.169.131.5", //Master IPVS worker node (Infra node 1 LB IP)
    "standby-director-ip": "192.169.131.7",//Backup IPVS worker node (Infra node 3 LB IP)
    "url_path": "",
    "status_code_expected": "200",
    "connect_timeout": "3",
    "nb_get_retry": "3",
    "delay_before_retry": "3"
  },
  "ipvs-service-account": {
    "sa_name": "ipvs-cluster-reader"
  },
  "openshfit-master-url": {
    "https-url": "https://cmt-osp-cluster.cmtlab-dns.com:8443/" //OpenShift Master URL
  }
}
```

Verifying Node Access

At this stage, use the ping command to verify that the following nodes are available on eth1:

- all infra nodes
- all worker nodes

Starting IPVS

To start the IPVS service (and pods):

Step 1 SSH into the deployer node as root.

Step 2 Navigate to the scripts directory.

```
cd /root/abr2ts-deployment/cisco-ipvs-os/deployment/scripts
```

Step 3 Use the following command to start the IPVS pods:

```
./k8s2ipvs.sh start -c ipvs_service_configure.json
```

Output

```
start IPVS service deployment
run generate_ipvs_service_files.sh...
Sun Jan 28 17:21:42 UTC 2018

Input params= ipvs_service_configure.json

====== deployment confgiure ======
ipvs service namespace           ipvs-service
ipvs pod cpu                     1
ipvs pod memory                  1Gi
ipvs node selector key           cisco.com/type
ipvs node selector value         master
ipvs backend selector key        cisco.com/type
ipvs backend selector value      backend
ipvs image
172.22.102.170:5000/cisco_ipvs_keepalived_os_release/ipvs_keepalived:latest
ipvs service account             ipvs-cluster-reader

====== IPVS confgiure ======
ipvs vip                         192.169.131.1
ipvs port                        80
ipvs network mask                255.255.255.0
ipvs active director IP          192.169.131.5
ipvs standby director IP         192.169.131.7
ipvs service namespace           abr2ts
ipvs serviceidentifier           vod-gateway
ipvs url path
ipvs status code expected        200
ipvs connect_timeout             3
ipvs nb_get_retry                3
ipvs delay_before_retry          3

finish generating configures
Sun Jan 28 17:21:42 UTC 2018
run deploy_ipvs_service.sh...
Sun Jan 28 17:21:42 UTC 2018

Input params= ipvs_service_configure.json

create namespace via cisco-ipvs-ns.yaml
namespace "ipvs-service" created
create service account via cisco-ipvs-ns.yaml
serviceaccount "ipvs-cluster-reader" created
cluster role "cluster-reader" added:
"system:serviceaccount:ipvs-service:ipvs-cluster-reader"
create configmap via cisco-ipvs-cm.yaml
configmap "ipvs-config" created
```

```
create daemonset via cisco-ipvs-ns.yaml
daemonset "ipvs-daemonset" created
Sun Jan 28 17:21:45 UTC 2018
run check_ipvs_service_running.sh...
Sun Jan 28 17:21:45 UTC 2018

Input params= ipvs_service_configure.json

Sun Jan 28 17:21:45 UTC 2018
check if namespace: ipvs-service is created
namespace ipvs-service created!
Sun Jan 28 17:21:45 UTC 2018
check if service account: ipvs-cluster-reader is created
service account ipvs-cluster-reader created!
Sun Jan 28 17:21:45 UTC 2018
check if configmap: ipvs-config is created
configmap ipvs-config created!
Sun Jan 28 17:21:45 UTC 2018
check if PODs in daemonset: ipvs-daemonset is created
Required:  2 Running:  0
check after sleep 5
daemonset ipvs-daemonset PODs created!
```

Note At the end of the output, a message should indicate that the pods have been started successfully

```
Sun Jan 28 17:21:51 UTC 2018
IPVS service deployed successfully
```

Verifying IPVS is Running

To verify that the IPVS pods are running you first verify that a new project has been created for IPVS. Next, you switch to that project, and lastly, you can use a couple of different commands to list the pods running in that project and other related information.

Step 1 Get a listing of the available OpenShift projects.

```
[root@cmt-deployer scripts]# oc get projects
```

Notice that a new project is listed for IPVS in the output.

Output

```
NAME               DISPLAY NAME   STATUS
abr2ts                            Active
default                           Active
infra                             Active
ipvs-service                      Active
kube-system                       Active
logging                           Active
management-infra                  Active
openshift                         Active
openshift-infra                   Active
```

Step 2 Switch to the new IPVS project.

```
[root@cmt-deployer scripts]# oc project ipvs-service
```

Output

```
Now using project "ipvs-service" on server "https://cmt-osp-cluster.cmtlab-dns.com:8443".
```

Step 3 List the running pods and provide information on any pods that are running.

```
[root@cmt-deployer scripts]# oc get pods -o wide
```

Output

```
NAME                    READY     STATUS    RESTARTS   AGE       IP              NODE
ipvs-daemonset-13bvd    1/1       Running   0          2m        172.22.102.65   cmt-infra3
ipvs-daemonset-ltlnx    1/1       Running   0          2m        172.22.102.58   cmt-infra1
```

Step 4 Verify the IPVS deployment status. The following command provides more detailed information than `oc get pods`. At the end of the output, you will see a test of "liveliness", which should show an OK status for the primary and backup pods. If that status is missing or different, then there is an issue with the pods.

If necessary, `cd /root/abr2ts-deployment/cisco-ipvs-os/deployment/scripts.`

Then run:

```
[root@cmt-deployer scripts]# ./k8s2ipvs.sh status -c ipvs_service_configure.json
```

Output

```
run check_ipvs_service_status.sh...
ipvs_service_configure.json
Sun Jan 28 17:27:35 UTC 2018

Input params= ipvs_service_configure.json


====== get POD deployment status ======
NAME                    READY     STATUS    RESTARTS   AGE       IP              NODE
ipvs-daemonset-13bvd    1/1       Running   0          5m        172.22.102.65   cmt-infra3
ipvs-daemonset-ltlnx    1/1       Running   0          5m        172.22.102.58   cmt-infra1

====== get IPVS status on active and backup IPVS director POD ======

checking POD: ipvs-daemonset-13bvd
status with connections:
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port               Conns   InPkts  OutPkts  InBytes OutBytes
  -> RemoteAddress:Port
TCP  192.169.131.1:80                    0        0        0        0        0
  -> 192.169.131.2:80                    0        0        0        0        0
  -> 192.169.131.3:80                    0        0        0        0        0
  -> 192.169.131.4:80                    0        0        0        0        0
status with weight:
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  192.169.131.1:80 wlc
  -> 192.169.131.2:80             Route   5      0          0
  -> 192.169.131.3:80             Route   5      0          0
  -> 192.169.131.4:80             Route   5      0          0

checking POD: ipvs-daemonset-ltlnx
status with connections:
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port               Conns   InPkts  OutPkts  InBytes OutBytes
  -> RemoteAddress:Port
TCP  192.169.131.1:80                    0        0        0        0        0
  -> 192.169.131.2:80                    0        0        0        0        0
  -> 192.169.131.3:80                    0        0        0        0        0
  -> 192.169.131.4:80                    0        0        0        0        0
status with weight:
```

```
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  192.169.131.1:80 wlc
  -> 192.169.131.2:80             Route   5      0          0
  -> 192.169.131.3:80             Route   5      0          0
  -> 192.169.131.4:80             Route   5      0          0


====== get liveness of IPVS director POD ======

checking agent at POD: ipvs-daemonset-13bvd
OK
checking agent at POD: ipvs-daemonset-ltlnx
OK
```

Step 5 Lastly, to verify that the IPVS VIP was added to eth1, execute the following:

```
# ip a | grep {IPVS VIP}
```

Sample output:

```
[root@cmt-infra1 ~]# ip a | grep 192.169.131.1
    inet 192.169.131.1/32 scope global eth1
```

Determining where IPVS Master is Running

To determine on which node the IVPS Master is running:

Step 1 SSH into the deployer node.

Step 2 Navigate to `/root/abr2ts-deployment/scripts`.

Step 3 Run the following OpenShift login command:

```
oc login -u system -p admin --insecure-skip-tls-verify=false
"https://cmt-osp-cluster.cmtlab-dns.com:8443" -n ipvs-service
```

Step 4 Run the following command:

```
./ipvs-master-info {LB VIP}
```

Command

```
[root@cmt-deployer scripts]# ./ipvs-master-info 172.22.102.244
```

Output

```
INFO: connecting to master-node: 172.22.102.244
IPVS Master-Node: cmt-infra1(172.22.102.58) VIP: 192.169.131.1
```

Stopping IPVS

To stop the IPVS pods:

Step 1 Run the command:

```
[root@ivpcoe-master1 scripts]# ./k8s2ipvs.sh stop -c ipvs_service_configure.json
```

Step 2 After about a minute, you can confirm that the IPVS service has stopped by typing:

```
oc get project.
```

Output

```
run remove_ipvs_service.sh...
../ipvs_service_configure.json
Wed Oct 11 19:13:01 UTC 2017

Input params= ../ipvs_service_configure.json

delete configmap...
configmap "ipvs-config" deleted
delete daemonset...
daemonset "ipvs-daemonset" deleted
delete pods...
No resources found
delete namespace...
namespace "ipvs-service" deleted
Checking if all pods are deleted...
No resources found.
Wed Oct 11 19:13:34 UTC 2017
All pods are deleted!
IPVS service stopped successfully
```

Running the Ingress Controller Tool

The Ingress Controller Tool is a deployer node tool that will adds ingress rules on both load balancers (Master and Standby) for any of the four services (OpenShift Master, Grafana, Prometheus, and Alert Manager) that are missing any rules.

The script will add these ports in HAProxy config so that we can access these dashboard from the Load Balancer VIP:

-- OpenShift Master: port ==> 8443

-- Grafana: port ==> 3000

-- AlertMgr: port ==> 9093

-- Prometheus: port ==> 9090

The following section explains how to run the Ingress Controller Tool and shows sample console output.

Step 1 If necessary, log into the deployer node as root.

Step 2 Change to the `scripts` directory.

```
cd /root/abr2ts-deployment/scripts
```

Step 3 Run the following command, and use the abr2ts-inventory file as an argument. Doing so will allow the tool to obtain information from the inventory file. The `-r yes` option will restart the haproxy application after updating its configuration.

```
[root@cmt-deployer scripts]# ./run_ingress_controller -f abr2ts-inventory -r yes
```

Output

```
INFO: HAProxy restart option: yes
--------------------------------
```

```
Good: File found: /root/ivp-coe/abr2ts-inventory
--------------------------------
#### Discovering MASTER Nodes from inventory file #####
FOUND: MASTER Node ==> cmt-master1 (172.22.102.143)
FOUND: MASTER Node ==> cmt-master2 (172.22.102.164)
FOUND: MASTER Node ==> cmt-master3 (172.22.102.169)
-------------------------------
Total MASTER Nodes: 3 #####
###############################
#### Discovering LB Nodes from inventory file #####
FOUND:     VIP ==> 172.22.102.244
FOUND: LB Node ==> cmt-lb1 (172.22.102.241)
FOUND: LB Node ==> cmt-lb2 (172.22.102.243)
-------------------------------
Total LB Nodes: 2 #####
###############################
INFO: connecting to lb-vip: 172.22.102.244
INFO: connection to lb-vip: 172.22.102.244 ... OK.
************** PLACING Grafana Block. ***************

frontend  atomic-grafana-openshift-api
    bind *:3000
    default_backend atomic-grafana-openshift-api
    mode tcp
    option tcplog

backend atomic-grafana-openshift-api
    balance source
    mode tcp
    server      master0 172.22.102.143:3000 check
    server      master1 172.22.102.164:3000 check
    server      master2 172.22.102.169:3000 check

************************************************
************** PLACING AlertMgr Block. **************

frontend  atomic-alertmanager-openshift-api
    bind *:9093
    default_backend atomic-alertmanager-openshift-api
    mode tcp
    option tcplog

backend atomic-alertmanager-openshift-api
    balance source
    mode tcp
    server      master0 172.22.102.143:9093 check
    server      master1 172.22.102.164:9093 check
    server      master2 172.22.102.169:9093 check

************************************************
************** PLACING Prometheus Block. ***************

frontend  atomic-prometheus-openshift-api
    bind *:9090
    default_backend atomic-prometheus-openshift-api
    mode tcp
    option tcplog

backend atomic-prometheus-openshift-api
    balance source
    mode tcp
    server      master0 172.22.102.143:9090 check
    server      master1 172.22.102.164:9090 check
    server      master2 172.22.102.169:9090 check
```

```
**************************************************
************** PLACING Grafana Block. ***************

frontend  atomic-grafana-openshift-api
    bind *:3000
    default_backend atomic-grafana-openshift-api
    mode tcp
    option tcplog

backend atomic-grafana-openshift-api
    balance source
    mode tcp
    server      master0 172.22.102.143:3000 check
    server      master1 172.22.102.164:3000 check
    server      master2 172.22.102.169:3000 check

**************************************************
************** PLACING AlertMgr Block. **************

frontend  atomic-alertmanager-openshift-api
    bind *:9093
    default_backend atomic-alertmanager-openshift-api
    mode tcp
    option tcplog

backend atomic-alertmanager-openshift-api
    balance source
    mode tcp
    server      master0 172.22.102.143:9093 check
    server      master1 172.22.102.164:9093 check
    server      master2 172.22.102.169:9093 check

**************************************************
************** PLACING Prometheus Block. ***************

frontend  atomic-prometheus-openshift-api
    bind *:9090
    default_backend atomic-prometheus-openshift-api
    mode tcp
    option tcplog

backend atomic-prometheus-openshift-api
    balance source
    mode tcp
    server      master0 172.22.102.143:9090 check
    server      master1 172.22.102.164:9090 check
    server      master2 172.22.102.169:9090 check

**************************************************
#### Listing all the Success results #####
SUCCESS: OCP Cluster reported masters: (cmt-master1 cmt-master2 cmt-master3), check ..OK.
SUCCESS: OCP-Master config for HAProxy: 172.22.102.241 is OK.
SUCCESS: Grafana config for HAProxy: 172.22.102.241 is OK.
SUCCESS: AlertManager config for HAProxy: 172.22.102.241 is OK.
SUCCESS: Prometheus config for HAProxy: 172.22.102.241 is OK.
SUCCESS: OCP-Master config for HAProxy: 172.22.102.243 is OK.
SUCCESS: Grafana config for HAProxy: 172.22.102.243 is OK.
SUCCESS: AlertManager config for HAProxy: 172.22.102.243 is OK.
SUCCESS: Prometheus config for HAProxy: 172.22.102.243 is OK.
SUCCESS: iptables OS_FIREWALL_ALLOW rule for OCP-Master: 172.22.102.241 is OK.
SUCCESS: iptables OS_FIREWALL_ALLOW rule for Grafana: 172.22.102.241 is OK.
SUCCESS: iptables OS_FIREWALL_ALLOW rule for AlertManager: 172.22.102.241 is OK.
SUCCESS: iptables OS_FIREWALL_ALLOW rule for Prometheus: 172.22.102.241 is OK.
```

```
SUCCESS: iptables OS_FIREWALL_ALLOW rule for OCP-Master: 172.22.102.243 is OK.
SUCCESS: iptables OS_FIREWALL_ALLOW rule for Grafana: 172.22.102.243 is OK.
SUCCESS: iptables OS_FIREWALL_ALLOW rule for AlertManager: 172.22.102.243 is OK.
SUCCESS: iptables OS_FIREWALL_ALLOW rule for Prometheus: 172.22.102.243 is OK.
SUCCESS: HAProxy restart: 172.22.102.241  ...OK.
SUCCESS: HAProxy restart: 172.22.102.243  ...OK.
```

Note The message at the end of the console output should indicate that no errors were found.

```
##############################################
#### Listing all the errors encountered #####
Great! NO Errors found. Total errors: 0 #####
##############################################
```

Monitoring Stack Overview

This section describes the process of installing the CMT monitoring stack, which consists of a Prometheus backend coupled with a Grafana user interface, AlertManager, and the Heapster cluster monitoring tool.

Prometheus is used to collect various metrics, such as network, memory, and CPU utilization, from the CMT cluster by scraping information from the endpoints. That information is stored locally so that rules can be run against it, or the data can be aggregated, if necessary.

Granafa provides a customizable dashboard user interface to view the node and cluster metrics collected by Prometheus.

Installing the Monitoring Stack

Prior to this Installing Prometheus and Grafana you should have uploaded the Docker images as per instructions earlier on in this document. To start the installation process:

Step 1 SSH as root into the deployer node.

Step 2 Log into OpenShift, using the master node IP address.

```
[root@ivpcoe-deployer ~]oc login -u system -p admin --insecure-skip-tls-verify=false
"https://cmt-osp-cluster.cmtlab-dns.com:8443"
```

Sample Output

```
Login successful.

You have access to the following projects and can switch between them with 'oc project
<projectname>':

abr2ts
* default
ipvs-service
ivp-logging
kube-system
logging
management-infra
openshift
openshift-infra
```

```
Using project "default".
[root@ivpcoe-deployer ~]#
```

Step 3 Switch to project ABR2TS.

```
[root@ivpcoe-deployer ~]# oc project abr2ts
```

Sample Output

```
Now using project "abr2ts" on server "https://172.22.98.80:8443".
```

Step 4 Navigate to the scripts directory.

```
#cd /root/abr2ts-deployment/scripts
```

Step 5 To configure the monitoring stack, execute the following:

```
[root@cmt-deployer scripts]# ./abr2ts_infra.sh config
```

Sample Output (truncated)

```
172.22.102.244
Kubernetes master is running at https://cmt-osp-cluster.cmtlab-dns.com:8443

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
Set context
Switched to context "abr2ts/cmt-osp-cluster-cmtlab-dns-com:8443/system".

Configuring abr2ts infra
Kubernetes master is running at https://cmt-osp-cluster.cmtlab-dns.com:8443

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
2018-01-28 17:39:02 Input params= /root/abr2ts-deployment
abr2ts/cmt-osp-cluster-cmtlab-dns-com:8443/system /root/.kube/config latest abr2ts_release
2018-01-28 17:39:02 updating abr2ts configs. kubeconfig=/root/.kube/config
2018-01-28 17:39:02 pic instance = abr2ts/cmt-osp-cluster-cmtlab-dns-com:8443/system
Context "abr2ts/cmt-osp-cluster-cmtlab-dns-com:8443/system" set.
clusterrole "prometheus" deleted
clusterrole "heapster" deleted
clusterrole "prometheus" created
serviceaccount "prometheus" created
clusterrolebinding "prometheus" created
clusterrole "heapster" created
serviceaccount "heapster" created
clusterrolebinding "heapster" created
2018-01-28 17:39:05 TAG= latest
2018-01-28 17:39:05 DR_GROUP= abr2ts_release

Starting abr2ts infra routes
Calling create_infra_route
2018-01-28 17:39:05 Creating Infra routes
2018-01-28 17:39:05 pic instance = abr2ts/cmt-osp-cluster-cmtlab-dns-com:8443/system
Context "abr2ts/cmt-osp-cluster-cmtlab-dns-com:8443/system" set.
2018-01-28 17:39:10 Creating prometheus route
route "prometheus" created
2018-01-28 17:39:10 Creating grafana route
route "grafana" created
2018-01-28 17:39:11 Creating alertmanager route
route "alertmanager" created
2018-01-28 17:39:11 Creating heapster route
route "heapster" created
cmt-infra1 cmt-infra2 cmt-infra3 cmt-master1 cmt-master2 cmt-master3 cmt-worker1
cmt-worker2 cmt-worker3
cmt-infra1
############################## WARNING!!! ###################################
################## READ THIS BEFORE ATTEMPTING TO LOGON ####################
```

```
#                                                                              #
#    This System is for the use of authorized users only.  Individuals    #
#    using this computer without authority, or in excess of their         #
#    authority, are subject to having all of their activities on this     #
#    system monitored and recorded by system personnel.  In the course    #
#    of monitoring individuals improperly using this system, or in the    #
#    course of system maintenance, the activities of authorized users     #
#    may also be monitored.  Anyone using this system expressly           #
#    consents to such monitoring and is advised that if such              #
#    monitoring reveals possible criminal activity, system personnel      #
#    may provide the evidence of such monitoring to law enforcement       #
#    officials.  You cannot copy, disclose, display or otherwise          #
#    communicate the contents of this server except to other Cisco        #
#    employees who have been authorized to access this server.            #
#                                                                              #
########################## Confidential Information ##########################
prometheus.yml
100% 7949     7.8KB/s   00:00
############################## WARNING!!! ###################################
################## READ THIS BEFORE ATTEMPTING TO LOGON ##################
#                                                                              #
#    This System is for the use of authorized users only.  Individuals    #
#    using this computer without authority, or in excess of their         #
#    authority, are subject to having all of their activities on this     #
#    system monitored and recorded by system personnel.  In the course    #
#    of monitoring individuals improperly using this system, or in the    #
#    course of system maintenance, the activities of authorized users     #
#    may also be monitored.  Anyone using this system expressly           #
#    consents to such monitoring and is advised that if such              #
#    monitoring reveals possible criminal activity, system personnel      #
#    may provide the evidence of such monitoring to law enforcement       #
#    officials.  You cannot copy, disclose, display or otherwise          #
#    communicate the contents of this server except to other Cisco        #
#    employees who have been authorized to access this server.            #
#                                                                              #
########################## Confidential Information ##########################
alert.rules
100% 2166     2.1KB/s   00:00
############################## WARNING!!! ###################################
################## READ THIS BEFORE ATTEMPTING TO LOGON ##################
#                                                                              #
#    This System is for the use of authorized users only.  Individuals    #
#    using this computer without authority, or in excess of their         #
#    authority, are subject to having all of their activities on this     #
#    system monitored and recorded by system personnel.  In the course    #
#    of monitoring individuals improperly using this system, or in the    #
#    course of system maintenance, the activities of authorized users     #
#    may also be monitored.  Anyone using this system expressly           #
#    consents to such monitoring and is advised that if such              #
#    monitoring reveals possible criminal activity, system personnel      #
#    may provide the evidence of such monitoring to law enforcement       #
#    officials.  You cannot copy, disclose, display or otherwise          #
#    communicate the contents of this server except to other Cisco        #
#    employees who have been authorized to access this server.            #
#                                                                              #
########################## Confidential Information ##########################
alertconfig.yml
100% 1752     1.7KB/s   00:00
cmt-infra2
The authenticity of host 'cmt-infra2 (172.22.102.61)' can't be established.
ECDSA key fingerprint is 1a:e1:89:1c:df:2c:b5:6a:8b:35:33:e6:a0:1d:eb:e0.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'cmt-infra2' (ECDSA) to the list of known hosts.
Connection closed by 172.22.102.61
```

```
lost connection
....
.... #Output is truncated at this point
....
alertconfig.yml
100% 1752     1.7KB/s   00:00
cmt-worker3
The authenticity of host 'cmt-worker3 (172.22.102.250)' can't be established.
ECDSA key fingerprint is d5:e8:c2:c4:81:1d:1a:96:6f:a1:01:9e:86:d5:98:b8.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'cmt-worker3' (ECDSA) to the list of known hosts.
############################### WARNING!!! ###################################
################## READ THIS BEFORE ATTEMPTING TO LOGON ###################
#                                                                         #
#    This System is for the use of authorized users only.  Individuals    #
#    using this computer without authority, or in excess of their         #
#    authority, are subject to having all of their activities on this     #
#    system monitored and recorded by system personnel.  In the course    #
#    of monitoring individuals improperly using this system, or in the    #
#    course of system maintenance, the activities of authorized users     #
#    may also be monitored.  Anyone using this system expressly           #
#    consents to such monitoring and is advised that if such              #
#    monitoring reveals possible criminal activity, system personnel      #
#    may provide the evidence of such monitoring to law enforcement       #
#    officials.  You cannot copy, disclose, display or otherwise          #
#    communicate the contents of this server except to other Cisco        #
#    employees who have been authorized to access this server.            #
#                                                                         #
######################### Confidential Information ########################
prometheus.yml
100% 7949     7.8KB/s   00:00
############################### WARNING!!! ###################################
################## READ THIS BEFORE ATTEMPTING TO LOGON ###################
#                                                                         #
#    This System is for the use of authorized users only.  Individuals    #
#    using this computer without authority, or in excess of their         #
#    authority, are subject to having all of their activities on this     #
#    system monitored and recorded by system personnel.  In the course    #
#    of monitoring individuals improperly using this system, or in the    #
#    course of system maintenance, the activities of authorized users     #
#    may also be monitored.  Anyone using this system expressly           #
#    consents to such monitoring and is advised that if such              #
#    monitoring reveals possible criminal activity, system personnel      #
#    may provide the evidence of such monitoring to law enforcement       #
#    officials.  You cannot copy, disclose, display or otherwise          #
#    communicate the contents of this server except to other Cisco        #
#    employees who have been authorized to access this server.            #
#                                                                         #
######################### Confidential Information ########################
alert.rules
100% 2166     2.1KB/s   00:00
############################### WARNING!!! ###################################
################## READ THIS BEFORE ATTEMPTING TO LOGON ###################
#                                                                         #
#    This System is for the use of authorized users only.  Individuals    #
#    using this computer without authority, or in excess of their         #
#    authority, are subject to having all of their activities on this     #
#    system monitored and recorded by system personnel.  In the course    #
#    of monitoring individuals improperly using this system, or in the    #
#    course of system maintenance, the activities of authorized users     #
#    may also be monitored.  Anyone using this system expressly           #
#    consents to such monitoring and is advised that if such              #
#    monitoring reveals possible criminal activity, system personnel      #
#    may provide the evidence of such monitoring to law enforcement       #
```

```
#    officials.  You cannot copy, disclose, display or otherwise          #
#    communicate the contents of this server except to other Cisco        #
#    employees who have been authorized to access this server.            #
#                                                                         #
########################## Confidential Information ########################
alertconfig.yml
100% 1752     1.7KB/s   00:00
[root@cmt-deployer scripts]#
```

Starting the Monitoring Stack

Step 1 To start Prometheus, Grafana, and Alert Manager services and pods, execute the following command:

```
[root@ivpcoe-deployer scripts]# ./abr2ts_infra.sh start
```

Sample output:

```
172.22.102.244
Kubernetes master is running at https://cmt-osp-cluster.cmtlab-dns.com:8443

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
Set context
Switched to context "abr2ts/cmt-osp-cluster-cmtlab-dns-com:8443/system".

Starting ABR2TS Infra services

Starting ABR2TS Infra pods
2018-01-28 17:55:27 starting abr2ts services. kubeconfig=/root/.kube/config
2018-01-28 17:55:27 pic instance = abr2ts/cmt-osp-cluster-cmtlab-dns-com:8443/system
Context "abr2ts/cmt-osp-cluster-cmtlab-dns-com:8443/system" set.
2018-01-28 17:55:27 checking pods. kubeconfig=/root/.kube/config
2018-01-28 17:55:27 pic instance = abr2ts/cmt-osp-cluster-cmtlab-dns-com:8443/system
Context "abr2ts/cmt-osp-cluster-cmtlab-dns-com:8443/system" set.
2018-01-28 17:55:38 Checking if all nodes are in ready state
2018-01-28 17:55:38 all 9 nodes are in ready state
2018-01-28 17:55:38 Starting prometheus rc
replicationcontroller "prometheus" created
2018-01-28 17:55:39 prometheus rc started
2018-01-28 17:55:39 Starting grafana rc
replicationcontroller "grafana" created
2018-01-28 17:55:39 grafana rc started
2018-01-28 17:55:39 Starting alertmanager rc
replicationcontroller "alertmanager" created
2018-01-28 17:55:39 alertmanager rc started
2018-01-28 17:55:39 Startging heapster rc
replicationcontroller "heapster" created
2018-01-28 17:55:40 heapster rc started
{"id":1,"message":"Datasource added","name":"abr2ts"}[root@cmt-deployer scripts]#
```

Step 2 Ensure that the Monitoring Stack services are properly running by executing the following command. If there are any problems, stop the services as shown in Stopping the Monitoring Stack, page 3-59.

Command: `[root@ivpcoe-deployer scripts]# oc get all`

Sample output:

```
NAME              DESIRED   CURRENT   READY     AGE
rc/alertmanager   1         1         1         8h
rc/grafana        1         1         1         8h
rc/heapster       1         1         1         8h
```

```
rc/prometheus     1         1         1         8h
rc/vod-gateway    20        20        20        9h

NAME                  HOST/PORT                          PATH      SERVICES       PORT
TERMINATION   WILDCARD
routes/alertmanager   alertmanager.abr2ts.cisco.com            alertmanager
<all>                   None
routes/grafana        grafana.abr2ts.cisco.com                 grafana
<all>                   None
routes/heapster       heapster.abr2ts.cisco.com                heapster
<all>                   None
routes/prometheus     prometheus.abr2ts.cisco.com              prometheus
<all>                   None

NAME               CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
svc/alertmanager   172.30.202.113   <nodes>       9093:9093/TCP    8h
svc/grafana        172.30.44.203    <nodes>       3000:3000/TCP    8h
svc/heapster       172.30.107.194   <nodes>       8082:8082/TCP    8h
svc/prometheus     172.30.144.134   <nodes>       9090:9090/TCP    8h
svc/vod-gateway    172.30.237.82    <nodes>       80:80/TCP        9h

NAME                     READY     STATUS    RESTARTS   AGE
po/alertmanager-p5qq2    1/1       Running   0          8h
po/fluent-269p7          1/1       Running   0          9h
po/fluent-87nvm          1/1       Running   0          9h
po/fluent-brmgm          1/1       Running   0          9h
po/fluent-c9m5h          1/1       Running   0          9h
po/fluent-gtjg6          1/1       Running   0          9h
po/fluent-l553x          1/1       Running   0          9h
po/fluent-rhjk6          1/1       Running   0          9h
po/fluent-tt1b8          1/1       Running   0          9h
po/fluent-z4976          1/1       Running   0          9h
po/fluent-z8kn9          1/1       Running   0          9h
po/grafana-mk6m6         1/1       Running   0          8h
po/heapster-8hwb8        1/1       Running   0          8h
po/prometheus-l85hh      1/1       Running   0          8h
po/vod-gateway-01xgf     1/1       Running   0          2h
po/vod-gateway-26lxr     1/1       Running   0          2h
po/vod-gateway-467vr     1/1       Running   0          2h
po/vod-gateway-7ml9f     1/1       Running   0          2h
po/vod-gateway-7p61p     1/1       Running   0          2h
po/vod-gateway-90b4t     1/1       Running   0          2h
po/vod-gateway-c9ptt     1/1       Running   0          2h
po/vod-gateway-cjccn     1/1       Running   0          2h
po/vod-gateway-cph95     1/1       Running   0          2h
po/vod-gateway-d6d1x     1/1       Running   0          2h
po/vod-gateway-d91c1     1/1       Running   0          2h
po/vod-gateway-gqn21     1/1       Running   0          2h
po/vod-gateway-mccdz     1/1       Running   0          2h
po/vod-gateway-mcggf     1/1       Running   0          2h
po/vod-gateway-qghsc     1/1       Running   0          2h
po/vod-gateway-rw13v     1/1       Running   0          2h
po/vod-gateway-sh161     1/1       Running   0          2h
po/vod-gateway-t133l     1/1       Running   0          2h
po/vod-gateway-tr6rg     1/1       Running   0          2h
po/vod-gateway-wd551     1/1       Running   0          2h
[root@cmt-deployer ~]#
```

Stopping the Monitoring Stack

To stop the monitoring stack, execute the following:

```
[root@ivpcoe-deployer scripts]# ./abr2ts_infra.sh stop
```

Sample output:

```
Kubernetes master is running at https://172.22.98.80:8443

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
2017-11-09 22:29:01 Setting abr2ts environment variables
2017-11-09 22:29:01 starting abr2ts services. kubeconfig=/root/.kube/config
2017-11-09 22:29:01 pic instance = abr2ts/172-22-98-80:8443/system
Set context
Switched to context "abr2ts/172-22-98-80:8443/system".
Stopping abr2ts infra
+ kubectl delete --grace-period=0 rc prometheus --namespace=abr2ts
+ kubectl delete --grace-period=0 rc grafana --namespace=abr2ts
+ kubectl delete --grace-period=0 pods,services -l app=prometheus --namespace=abr2ts
+ kubectl delete --grace-period=0 pods,services -l app=grafana --namespace=abr2ts
+ kubectl delete --grace-period=0 ep -l app=prometheus --namespace=abr2ts
+ kubectl delete --grace-period=0 ep -l app=grafana --namespace=abr2ts

+ set +x
```

Verifying the Cluster

At this point, the installation process for the CMT VoD Gateway is complete. The next step is to verify the cluster.

Step 1 If necessary, SSH into the deployer node.

Step 2 Navigate to the scripts folder.

```
cd /root/abr2ts-deployment/scripts
```

Step 3 Run the following cluster verification command:

```
# ./verify-cluster-configuration -m <LB VIP> -u system -p admin
```

Output

```
INFO: connecting to master-node: 172.22.102.244
#### Verifying Backend Nodes through Labels #####
FOUND: Backend Node ==> cmt-worker1
FOUND: Backend Node ==> cmt-worker2
FOUND: Backend Node ==> cmt-worker3
------------------------------
Total Backend Nodes: 3 #####
##############################
#### Verifying IPVS Nodes through Labels #####
FOUND: IPVS Node ==> cmt-infra1
FOUND: IPVS Node ==> cmt-infra3
------------------------------
Total IPVS Nodes: 2 #####
##############################
#### Verifying INFRA Nodes through Labels #####
FOUND: INFRA Node ==> cmt-infra1
FOUND: INFRA Node ==> cmt-infra2
FOUND: INFRA Node ==> cmt-infra3
------------------------------
Total INFRA Nodes: 3 #####
##############################
#### Listing all the Success results #####
```

```
SUCCESS: All IPVS Nodes are found: Count: 2 ...OK
SUCCESS: All INFRA Nodes are found: Count: 3 ...OK
SUCCESS: lo:1 vip: cmt-worker1 is OK.
SUCCESS: lo:1 vip: cmt-worker2 is OK.
SUCCESS: lo:1 vip: cmt-worker3 is OK.
SUCCESS: SYSCTL: cmt-worker1 is OK.
SUCCESS: SYSCTL: cmt-worker2 is OK.
SUCCESS: SYSCTL: cmt-worker3 is OK.
SUCCESS: SYSCTL: cmt-infra1 is OK.
SUCCESS: SYSCTL: cmt-infra3 is OK.
SUCCESS: Network Labels: cmt-worker1 is OK.
SUCCESS: Network Labels: cmt-worker2 is OK.
SUCCESS: Network Labels: cmt-worker3 is OK.
SUCCESS: Network Labels: cmt-infra1 is OK.
SUCCESS: Network Labels: cmt-infra3 is OK.
SUCCESS: iptables OS_FIREWALL_ALLOW rule: cmt-infra1 is OK.
SUCCESS: iptables OS_FIREWALL_ALLOW rule: cmt-infra3 is OK.
SUCCESS: DNS config for CDN(svr): cmt-worker1 is OK.
SUCCESS: DNS config for CDN(ttl): cmt-worker1 is OK.
SUCCESS: DNS config for CDN(svr): cmt-worker2 is OK.
SUCCESS: DNS config for CDN(ttl): cmt-worker2 is OK.
SUCCESS: DNS config for CDN(svr): cmt-worker3 is OK.
SUCCESS: DNS config for CDN(ttl): cmt-worker3 is OK.
##############################################
#### Listing all the errors encountered #####
Great! NO Errors found. Total errors: 0 #####
##############################################
[root@cmt-deployer scripts]#
```

Configuring Grafana

The following section details the steps that are required to configure the Grafana interface for use with the reference dashboards provided by Cisco.

Step 1 Edit the `/etc/hosts` file on the machine from which you will be accessing the Grafana user interface. Add the Grafana hostname to the load balancer VIP IP. For example:

```
##
# Host Database
#
# localhost is used to configure the loopback interface
# when the system is booting.  Do not change this entry.
##
127.0.0.1localhost
255.255.255.255broadcasthost
::1             localhost

172.22.102.244 grafana.abr2ts.cisco.com  ' Load Balancer VIP IP
172.22.102.244 alertmanager.abr2ts.cisco.com  ' Load Balancer VIP IP
172.22.102.244 prometheus.abr2ts.cisco.com  ' Load Balancer VIP IP
```

Step 2 Using the previous hostname setting as an example, log into the Grafana interface on port 3000. The credentials are `username: admin` / `password: admin`.

```
http://grafana.abr2ts.cisco.com:3000
```

Step 3 Navigate to **Add data source**.

Step 4 Enter the following values onto the data source configuration page.

Table 3-1 Add/Edit Data Source

Field	Value
Name	abr2ts
Type	Prometheus
URL	http://<LB VIP>:9090

Importing Grafana Dashboards

The following procedures will import the Grafana Dashboards, allowing you to monitor metrics for the Kubernetes cluster and for the Worker nodes.

Step 1 Copy the `Media-Transformer-Workers-Dashboard.json` and `Media-Transformer-Cluster-Monitoring.json` files to the localhost where you are opening the Grafana user interface. These files will need to be importing to create the Grafana dashboards. The json files are located on the Deployer node at:
`/root/abr2ts-deployment/platform/resources/config/grafana`

Note Whenever you restart the Monitoring Stack, you will need to re-import the `Media-Transformer-Workers-Dashboard.json` and `Media-Transformer-Cluster-Monitoring.json` files in order to view the dashboards again.

Step 2 Navigate to **Dashboards > Import.**

Step 3 Import the `Media-Transformer-Workers-Dashboard.json`.

Step 4 Select "`abr2ts`" as the Prometheus data source.

Step 5 Verify that the dashboard shows all of the CMT pod data, such as: transmit/receive/memory/CPU usage.

Step 6 Navigate to **Dashboards > Import** once again.

Step 7 Import the `Media-Transformer-Cluster-Monitoring.json` file to the dashboard.

Step 8 Select "`abr2ts`" as the Prometheus data source.

Step 9 Verify that the dashboard shows cluster node metrics, such as network I/O, memory, CPU, and filesystem usage.

Adding Routes for Infra & Worker Nodes

Proper routes need to be added to Worker and Infra nodes so that they can communicate with the VDS-TV streamers and the content delivery network (CDN). Shown below are sample routes for the nodes.

Routes for CDN (Worker only)

```
ip route add 192.169.130.0/24 via 192.169.150.246 dev eth1
ip route add 192.169.131.0/24 via 192.169.150.246 dev eth1
```

```
ip route add 192.169.132.0/24 via 192.169.150.246 dev eth1
ip route add 192.169.133.0/24 via 192.169.150.246 dev eth1
```

Routes for Streamer (Worker and Infra)

```
ip route add 192.169.125.0/24 via 192.169.150.246 dev eth1
```

Ingesting & Streaming Content

The following section provides instructions on ingesting and streaming ABR content.

Provisioning ABR Content

Step 1 SSH as root into the VDS-TV Master Vault.

Step 2 Verify that the CMT IPVS VIP is mapped to the hostname and set within `/etc/hosts`.

`For example: <IPVS VIP>  hostname`

Step 3 Verify that the CMT IPVS VIP can be reached (pinged).

Figure A-1 Pinging the IPVS VIP

```
[root@VM-SSV-209 ~]# cat /etc/hosts
# Do not remove the following line, or various programs
# that require network functionality will fail.
# Lines below this one modified by cdsnetconfig
127.0.0.1               localhost.localdomain localhost
::1             localhost6.localdomain6 localhost6
172.22.98.30    abr2ts-vgserver.com
10.197.86.209   VM-SSV-209
[root@VM-SSV-209 ~]# ping abr2ts-vgserver.com
PING abr2ts-vgserver.com (172.22.98.30) 56(84) bytes of data.
64 bytes from abr2ts-vgserver.com (172.22.98.30): icmp_seq=1 ttl=46 time=238 ms
64 bytes from abr2ts-vgserver.com (172.22.98.30): icmp_seq=2 ttl=46 time=237 ms
^C
--- abr2ts-vgserver.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 237.771/237.934/238.097/0.163 ms
```

Step 4 Add a proper route to the IPVS VIP network via the Vault Ingest network.

Step 5 Add an ingest network route to all Infra and Worker Nodes.

Step 6 Login as user `isa`.

Step 7 Change directory to the `IntegrationTest` folder.

```
cd /home/isa/IntegrationTest
```

Step 8 Execute the following script:

```
./list_all_contents
```

Figure A-2 Ingest - List_all-contents

```
[isa@VM-SSV-209 ~/IntegrationTest]$ ./list_all_contents

*************** LIST OF CONTENTS **********************

Factory Name    : ArroyoContentStoreFactory
OperationaState: os_InService
AdminState      : as_InService
Servant Name    : ArroyoVideoContentStore
OperationaState: os_InService
AdminState      : as_InService

****************** END OF LIST ************************

[isa@VM-SSV-209 ~/IntegrationTest]$
```

Step 9 Change to the `client` directory.

```
cd /home/isa/ContentStore/client
```

Step 10 Verify the following parameters within the `provision_content` script:

- NAME_SERVICE_HOST —> is the Name Server IP
- NAME_SERVICE_PORT —> is the Name Server port
- VideoContentStore —> Should be the "Content Store Name" value as given on the **Configure > Array Level > Vault BMS** (Business Management Services) page.

Figure A-3 Provision Content Script

```
#!/bin/bash
##------------------------------------------------------------------------
##        This document contains material which is the proprietary
##   property of and confidential to CALYPSO. Disclosure outside CALYPSO is
## prohibited except by license agreement or other confidentiality agreement.
##
##                    Copyright (c) 2002 Calypso Networks, LLC
##                    Copyright (c) 2017 by Cisco Systems, Inc.
##                         All rights reserved.
##------------------------------------------------------------------------
##
##   File Name      : provision_content
##   Creation Date  : Tue Jul 15 13:44:54 PDT 2003
##   Author         : Neeraj K Sharma
##   Description    : This script executes some test on
##                    CalypsoContentStoreFatory.
##------------------------------------------------------------------------

export NAME_SERVICE_HOST=10.197.86.209
export NAME_SERVICE_PORT=2000

export VideoContentStore=ArroyoVideoContentStore
```

Step 11 Run a command that uses the `provision_content` script to ingest CMT content:

```
# ./provision_content  <ProvideID-AssetID-ContentName> <CMT URL>
```

Figure A-4 Running the Provision Content Script - Desired Output

```
[isa@VM-SSV-209 client]$ ./provision_content ABR2TS.com-Sampleingest-test1.mpd http://abr2ts-vgserver.com/sample/manifest.mpd

******* STARTING ARROYO CONTENT INGEST CLIENT:

Ingesting Content (ABR2TS.com-Sampleingest-test1.mpd)..... DONE

********** RESULT **************************

PROVISION RESULT: Content(ABR2TS.com-Sampleingest-test1.mpd) ingested successfully.

******************************************************

[isa@VM-SSV-209 client]$
```

Verifying Ingestion Status

The following section describes how to verify the ingestion status on VDS-TV and the CMT pod logs.

Step 1 Check the following log for the ingest status on the Master vault.

`/arroyo/log/ContentStoreMaster.log`

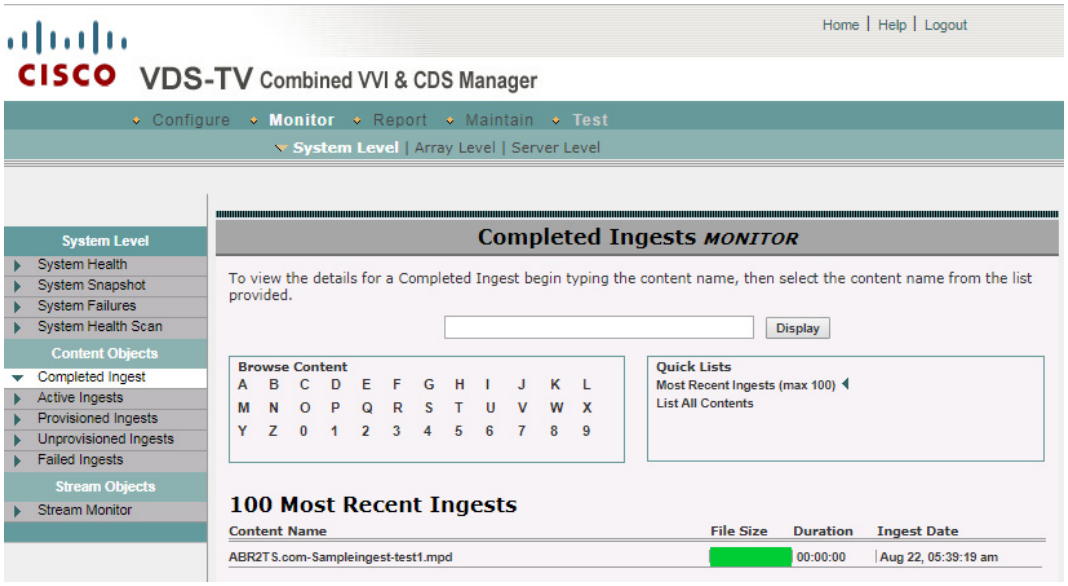
Figure A-5 Content Store Master Log

```
[isa@VM-SSV-209 client]$ tail -23 /arroyo/log/ContentStoreMaster.log
2017-08-22 12:39:19.428173 UTC localhost ContentStoreImpl.cc879In  createServant:=ABR2TS.com-Sampleingest-test1.mpd
2017-08-22 12:39:19.428623 UTC localhost ContentStoreImpl.cc959Persisting Content Object:=ABR2TS.com-Sampleingest-test1.mpd
2017-08-22 12:39:19.429452 UTC localhost ContentStoreImpl.cc|981|Persisting Content Object....done
2017-08-22 12:39:19.450539 UTC localhost ContentStoreImpl.cc1067Out createServant:=ABR2TS.com-Sampleingest-test1.mpd
2017-08-22 12:39:19.534687 UTC localhost ContentImpl.cc797In  provision:=ABR2TS.com-Sampleingest-test1.mpd::ArroyoVideoContentStore
2017-08-22 12:39:19.535343 UTC localhost ContentImpl.cc|1672|CDNCONTENT(ATIS or ABR), and only basic field in content object table update
2017-08-22 12:39:19.537173 UTC localhost ContentImpl.cc1747A valid host name found in the http URL:=abr2ts-vgserver.com
2017-08-22 12:39:19.537309 UTC localhost ContentImpl.cc1776Domain name (abr2ts-vgserver.com) resolved to an address with octets count(h_length):4
2017-08-22 12:39:19.537374 UTC localhost ContentImpl.cc1782Host Name (abr2ts-vgserver.com) resolved to IP Address:=172.22.98.30
2017-08-22 12:39:19.537411 UTC localhost ABRClient.cc141<INFO> ABR2TS client initialied with MPD URL: http://abr2ts-vgserver.com/sample/manifest.mpd IP: 172.22.98.30 PORT:
 80 Host: abr2ts-vgserver.com
2017-08-22 12:39:19.537439 UTC localhost ABRClient.cc149<INFO> Connecting to ABR2TS IP: 172.22.98.30 PORT:80
2017-08-22 12:39:19.777629 UTC localhost ABRClient.cc183<INFO> Successfully established connection to ABR2TS IP: 172.22.98.30 PORT:80
2017-08-22 12:39:19.777667 UTC localhost ABRClient.cc|216|<INFO> Invoking bitrate API: GET /bitrate
2017-08-22 12:39:20.523176 UTC localhost ABRClient.cc245<INFO>bitrate API .....DONE. Response Received:=HTTP/1.1 200 OK
Cbr-Bitrate: 7619917
Date: Tue, 22 Aug 2017 05:34:44 GMT
Content-Length: 0
Content-Type: text/plain; charset=utf-8

2017-08-22 12:39:20.523232 UTC localhost ABRClient.cc325Rate:=7619917
2017-08-22 12:39:20.523253 UTC localhost DbContentObject2K.cc283In DbContentObject2K::updateIt::mName=ABR2TS.com-Sampleingest-test1.mpd
2017-08-22 12:39:20.531465 UTC localhost DbContentObject2K.cc325Out DbContentObject2K::updateIt::mName=ABR2TS.com-Sampleingest-test1.mpd
2017-08-22 12:39:20.531682 UTC localhost ContentImpl.cc1853Out provision :=ABR2TS.com-Sampleingest-test1.mpd::ArroyoVideoContentStore
[isa@VM-SSV-209 client]$
```

Step 2 Check the **Completed Ingest** page in VDS-TV interface (combined with VVI & CDS Manager) for the status of the Ingest operation.

Figure A-6 VDS-TV



Step 3 Now we will check the VOD Gateway log. First, you must log into the Deployer node as root.

Step 4 Make abr2ts the current project.

```
oc project abr2ts
```

Step 5 Navigate to the scripts folder.

```
cd /root/abr2ts-deployment/scripts
```

Step 6 Run the following command to tail the VOD Gateway logs from all the pods.

```
./kubetail.sh vod-gateway
```

Streaming ABR Content

Step 1 SSH into the Streamer.

Step 2 Verify the IPVS VIP hostname in `/etc/hosts`.

Step 3 Verify that the IPVS VIP hostname can be reached (pinged).

Note If SSV is used, then prior to streaming we must change from the worker node IP to the IPVS VIP and we have to execute the following command: `echo 1 > /proc/calypso/tunables/read_etc_hosts`

Step 4 Login as user `isa`.

Step 5 Change directory to the `IntegrationTest` folder.

```
cd /home/isa/IntegrationTest
```

Step 6 Execute the following script:

```
./list_all_streams
```

Figure A-7 Running the list_all_streams script

```
[isa@VM-SSV-209 ~/IntegrationTest]$ ./list_all_streams

**************************************************

StreamService: ArroyoStreamService

*************** LIST OF STREAMS **********************


****************** END OF LIST ************************

[isa@VM-SSV-209 ~/IntegrationTest]$
```

Step 7 Change directory to the `client` folder.

```
cd /home/isa/Streaming/client
```

Step 8 Ensure that the following parameters are configured in `CalypsoStreamClient.cfg`

- DestinationIPAddress —> should be the destination IP configured at: **Configure > System Level > QAM Gateway**
- DestinationPortNumber —> `GigePorts.txt` should be updated with the port number. For example: `[isa@str240_mkt1 client]$ cat GigePorts.txt 1001`
- NSGServiceGroup —> should be the service group number

Figure A-8 CalypsoStreamService.cfg

```
#-----------------------------------------------------------------------
#
#   File Name      : CalypsoStreamService.cfg
#   Creation Date  : Tue Mar 25 10:29:26 PST 2003
#   Author         : Neeraj K Sharma
#   Description    : This files provides all Calypso StreamService server
#                    related configuration parameters as key-value pair.
#                    The comments should start with pound (#) sign.
##-----------------------------------------------------------------------
CalypsoStreamService  = ArroyoStreamService_virt
ContentStoreFactoryId = ArroyoContentStoreFactory_virt
ContentStoreId        = ArroyoVideoContentStore_virt
CalypsoSession        = C0A800F0DB2A::12345
EventChannels         = EventChannels
Context               = Context
ChannelId             = ContentChannel
ChannelKind           = Channel

#Process Params:
NumberOfThreads = 25
NumberOfStreams = 3000

#Play Stream:
PlayStream      = 0

#Session setup params:
CreateStream    = 1
ProvisionStream = 0

#SessionNegotiation Callback Params:
GigePortsFile         = GigePorts.txt
DestinationIPAddress  = 192.168.2.42
BlackHoleIPAddress    = 192.168.2.42

# Service Group for ASI:
NSGServiceGroup       = 1
BlackHoleServiceGroup = 1
```

Step 9 Execute a script to setup and play a stream.

```
run_client
```

Step 10 Type the following options:

```
1 > 34 > 1 > y > <ProviderID-AssetID-Contentname>
```

Figure A-9 Options for run_client (1 of 2)

```
[isa@VM-SSV-209 client]$ ./run_client
Waiting for SessionThread to get ready......
CORBA EXCEPTION : user exception, ID 'IDL:isa.twc.com/ServerModule/NameNotFound:1.0'
No Content found inservice specified in the config file.
**************************************

       1: CalypsoStreamService
       2: Stream
       0: EXIT

Enter [1/2/3/0]? : 1
****************************
StreamService: ArroyoStreamService
****************************
```

Figure A-10 Options for run_client (2 of 2)

```
  34: ProvisionAndPlay
  35: CreateProvisionAndPlay
  36: CreateStreamWithPGN
  37: DisplayContentRefCount
  38: CreatePVRStream
  39: ReloadServiceGroups
  40: CreateStreamMultipleRanged
  41: CreateVODStream
  42: ****
  43: GetTrickmodeEvents
  44: CreateRangedPlaylistStream
  45: CreateStreamFromPlaylist
  50: Run Soak Test
  ..............................
  ..............................
  91: Test Corrupt Session
   0: EXIT

Enter [1/2/3/../0]? : 34
Enter HowMany  : 1
Send to NSG [Y|y]:y
Enter Content Name:ABR2TS.com-Sampleingest-test1.mpd
```

Note To verify the stream state, check the following log on the VDS-TV streamer:
`/arroyo/log/Protocoltiming.log.<date>`

Step 11 Next, we will check the VOD Gateway log. First, log into the Deployer node as root.

Step 12 Make abr2ts the current project.

```
oc project abr2ts
```

Step 13 Navigate to the scripts folder.

```
cd /root/abr2ts-deployment/scripts
```

Step 14 Run the following command to tail the VOD Gateway logs from all the pods.

```
./kubetail.sh vod-gateway
```

Heapster Logs

Heapster Overview

Heapster gathers the metrics from across the OpenShift cluster. It retrieves metadata associated with the cluster from the master API and retrieves individual metrics from the /stats endpoint which is exposed on each individual OpenShift node.It gathers system level metrics such as CPU, Memory, Network, etc

Aggregates

The metrics are initially collected for nodes and containers and later aggregated for pods, namespaces and clusters. Disk and network metrics are not available at container level (only at pod and node level).

Heapster exports the following metrics to its backends.

Table B-1 Exported Heapster Metrics

Metric Name	Description
cpu/limit	CPU hard limit in millicores.
cpu/node_capacity	CPU capacity of a node.
cpu/node_allocatable	CPU allocatable of a node.
cpu/node_reservation	Share of cpu that is reserved on the node allocatable.
cpu/node_utilization	CPU utilization as a share of node allocatable.
cpu/request	CPU request (the guaranteed amount of resources) in millicores.
cpu/usage	Cumulative CPU usage on all cores.
cpu/usage_rate	CPU usage on all cores in millicores.
filesystem/usage	Total number of bytes consumed on a filesystem.
filesystem/limit	The total size of filesystem in bytes.
filesystem/available	The number of available bytes remaining in a the filesystem
filesystem/inodes	The number of available inodes in a the filesystem
filesystem/inodes_free	The number of free inodes remaining in a the filesystem
disk/io_read_bytes	Number of bytes read from a disk partition

Table B-1 Exported Heapster Metrics

Metric Name	Description
disk/io_write_bytes	Number of bytes written to a disk partition
disk/io_read_bytes_rate	Number of bytes read from a disk partition per second
disk/io_write_bytes_rate	Number of bytes written to a disk partition per second
memory/limit	Memory hard limit in bytes.
memory/major_page_faults	Number of major page faults.
memory/major_page_faults_rate	Number of major page faults per second.
memory/node_capacity	Memory capacity of a node.
memory/node_allocatable	Memory allocatable of a node.
memory/node_reservation	Share of memory that is reserved on the node allocatable.
memory/node_utilization	Memory utilization as a share of memory allocatable.
memory/page_faults	Number of page faults.
memory/page_faults_rate	Number of page faults per second.
memory/request	Memory request (the guaranteed amount of resources) in bytes.
memory/usage	Total memory usage.
memory/cache	Cache memory usage.
memory/rss	RSS memory usage.
memory/working_set	Total working set usage. Working set is the memory being used and not easily dropped by the kernel.
accelerator/memory_total	Memory capacity of an accelerator.
accelerator/memory_used	Memory used of an accelerator.
accelerator/duty_cycle	Duty cycle of an accelerator.
network/rx	Cumulative number of bytes received over the network.
network/rx_errors	Cumulative number of errors while receiving over the network.
network/rx_errors_rate	Number of errors while receiving over the network per second.
network/rx_rate	Number of bytes received over the network per second.
network/tx	Cumulative number of bytes sent over the network
network/tx_errors	Cumulative number of errors while sending over the network
network/tx_errors_rate	Number of errors while sending over the netwo
network/tx_rate	Number of bytes sent over the network per second.
uptime	Number of milliseconds since the container was started.

Figure B-1 Sample Metrics from Splunk

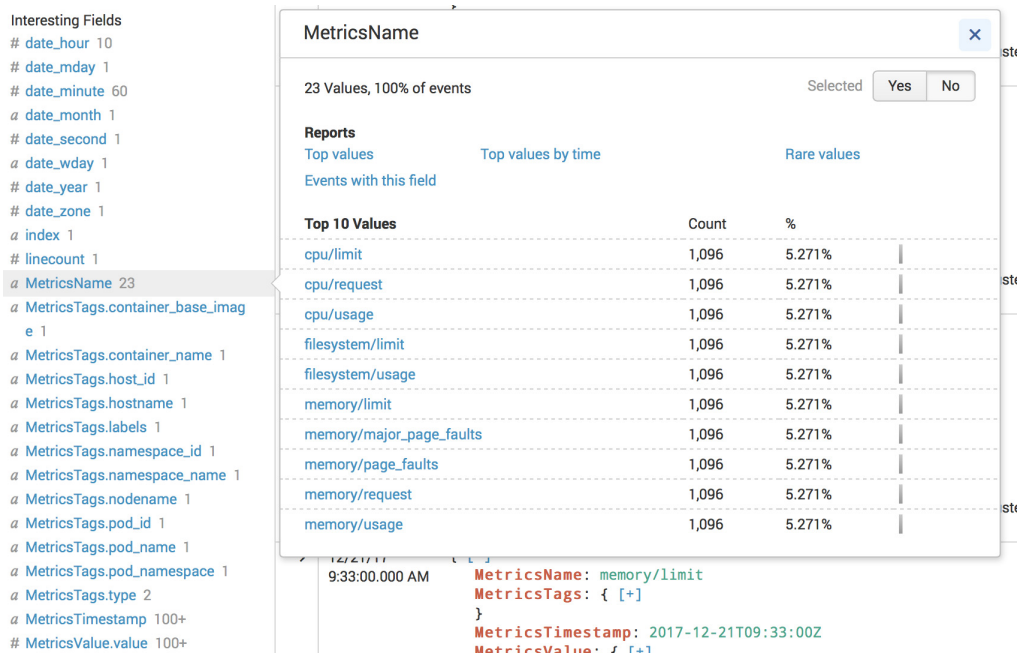


Table B-2 Heapster Labels

Label Name	Description
pod_id	Unique ID of a Pod
pod_name	User-provided name of a Pod
container_base_image	Base image for the container
container_name	User-provided name of the container or full cgroup name for system containers
host_id	Cloud-provider specified or user specified Identifier of a node
hostname	Hostname where the container ran
nodename	Nodename where the container ran
labels	Comma-separated (default) list of user-provided labels. Format is 'key:value'
namespace_id	UID of the namespace of a Pod
namespace_name	User-provided name of a Namespace
resource_id	A unique identifier used to differentiate multiple metrics of the same type. e.x. Fs partitions under filesystem/usage, disk device name under disk/io_read_bytes
make	Make of the accelerator (nvidia, amd, google etc.)
model	Model of the accelerator (tesla-p100, tesla-k80 etc.)
accelerator_id	ID of the accelerator

Alert Rules

Alert Rules Overview

Prometheus allows users to define alert conditions based upon predefined expressions within an Alert Rules file. It then notifies an external service (AlertManager in our case) to fire alerts once specific thresholds have been reached. Whenever an alert expression is evaluated as true, that alert becomes active.

Updating Alert Rules

The following process is used to update the Alert Rules file, update necessary configuration settings, and then restart the system so that the changes take effect. To update Alert Rules:

Step 1 If necessary, SSH as root into the Deployer node.

Step 2 The rules file is located at:
`/root/abr2ts-deployment/platform/resources/config/prometheus/alert.rules`

Step 3 Make a backup of the rules file.

Step 4 Edit the file to set the parameters and thresholds that you need monitored.
For background information on the Prometheus querying language, rules, conventions, and available metrics, see Alert Rules Reference Materials, page C-2.

Step 5 Navigate to `/root/abr2ts-deployment/scripts/`

Step 6 Run this command to stop the Infra node.

```
./abr2ts_infra.sh stop
```

Step 7 Run this command to update the Alert Manager configuration settings on the Infra node.

```
./abr2ts_infra.sh config
```

Step 8 Run this command to start the Infra node. The rules file will automatically be loaded at start up.

```
./abr2ts_infra.sh start
```

Alert Rules Reference Materials

The following section provides links to background information that you will find useful when creating or editing Alert Rules:

- For details on how Alert Rules are defined, refer to:
  `https://prometheus.io/docs/prometheus/latest/configuration/alerting_rules/`
- For details on the Prometheus querying language, refer to:
  `https://prometheus.io/docs/prometheus/latest/querying/basics/`
- Metrics probed by the querying functions are provided by the Kubernetes API. Information related to metrics and monitoring is available at this URL:
  `https://coreos.com/blog/monitoring-kubernetes-with-prometheus.html`

Sample Alert Rule

The following section lists a sample Alert Rule for your reference.

Sample Alert Rule

```
ALERT ClusterContainerMemoryUsage

  IF sum (container_memory_working_set_bytes{id="/",kubernetes_io_hostname=~"abr2ts-.*"})
/ sum (machine_memory_bytes{kubernetes_io_hostname=~"abr2ts-.*"}) * 100  > 50

  FOR 10s

  LABELS { severity = "critical" }

  ANNOTATIONS {

     summary = "cluster containers consuming high level of memory",

     description = ""

  }
```

Alert Rule Commands

The following table provides explanations for some commands used when creating Alert Rules.

Table C-1 Alert Rule Commands

Label	Possible Values	Description
ALERT	QueryContainerMemoryUsage	Name of alert rule
ANNOTATIONS	summary = “…….” Description = “…….”	Annotations for the alert
FOR	10s	The optional for clause causes Prometheus to wait for a certain duration between first encountering a new matching condition.

Table C-1 Alert Rule Commands

Label	Possible Values	Description
IF	sum(container_memory_working_set_bytes{id="/",kubernetes_io_hostname=~"abr2ts-.*"}) / sum (machine_memory_bytes{kubernetes_io_hostname=~"abr2ts-.*"}) * 100 > 90	sum is a Prometheus query function container_memory_working_set_bytes and machine_memory_bytes are kubernetes metrics expression checks whether memory usage is greater than 90%
LABELS	severity=”critical”	One or more labels for the alert

Inspecting Alerts at Runtime

To manually view the exact label sets for which alerts are active (meaning pending or firing), navigate to the "Alerts" tab within Prometheus. The alert value is set to 1 as long as the alert remains in an active state. When the alert transitions to an inactive state, the alert value will be changed to 0 by the system.

Sending Alert Notifications

Prometheus' Alert Rules are suitable for basically assessing what is going wrong at a given time. An additional component is required to add summarization, notification rate limiting, silencing, and other features on top of the provided simple alert definitions. The AlertManager component takes on this task. Prometheus is configured to periodically send information about alert states to the AlertManager instance, which is then responsible for dispatching the right notifications. Figure C-1 on page C-4 and Figure C-2 depict pending and firing alerts as shown in AlertManager.

Figure C-1 Alert Manager UI - pending alerts

Figure C-2 Alert Manager UI - firing alerts

Sample Alert Notifications

The following default sample alerts will be packaged with the CMT release.

Table C-2 Sample Alert Notifications

Label	Description	Default Duration
NodeDown	A node in Media Transformer is down for n minutes	5 minutes
VODGatewayTotalMemoryUsage	VOD Gateway memory usage exceeded a certain threshold on a node.	10 minutes
VODGatewayPercentageMemory Usage	VOD Gateway node memory usage exceeded a threshold percentage (default=90%)	10 minutes
VODGatewayCPUUsage	VOD Gateway node CPU usage exceeded a threshold percentage (default=80%)	10 minutes
ClusterContainerMemoryUsage	Overall memory usage of containers in the cluster exceeded a certain threshold percentage (default=90%)	10 minutes
ApiServerDown	The Kubernetes server is down. This indicates that the system is probably unstable.	5 minutes