Avid MediaCentral Platform Services

Concepts and Clustering Guide

Legal Notices

Product specifications are subject to change without notice and do not represent a commitment on the part of Avid Technology, Inc.

This product is subject to the terms and conditions of a software license agreement provided with the software. The product may only be used in accordance with the license agreement.

This product may be protected by one or more U.S. and non-U.S patents. Details are available at www.avid.com/patents.

This guide is protected by copyright. This guide is for your personal use and may not be reproduced or distributed, in whole or in part, without permission of Avid. Reasonable care has been taken in preparing this guide; however, it may contain omissions, technical inaccuracies, or typographical errors. Avid Technology, Inc. disclaims liability for all losses incurred through the use of this document. Product specifications are subject to change without notice.

The following disclaimer is required by Apple Computer, Inc.:

APPLE COMPUTER, INC. MAKES NO WARRANTIES WHATSOEVER, EITHER EXPRESS OR IMPLIED, REGARDING THIS PRODUCT, INCLUDING WARRANTIES WITH RESPECT TO ITS MERCHANTABILITY OR ITS FITNESS FOR ANY PARTICULAR PURPOSE. THE EXCLUSION OF IMPLIED WARRANTIES IS NOT PERMITTED BY SOME STATES. THE ABOVE EXCLUSION MAY NOT APPLY TO YOU. THIS WARRANTY PROVIDES YOU WITH SPECIFIC LEGAL RIGHTS. THERE MAY BE OTHER RIGHTS THAT YOU MAY HAVE WHICH VARY FROM STATE TO STATE.

The following disclaimer is required by Sam Leffler and Silicon Graphics, Inc. for the use of their TIFF library:

Copyright © 1988–1997 Sam Leffler
Copyright © 1991–1997 Silicon Graphics, Inc.

Permission to use, copy, modify, distribute, and sell this software [i.e., the TIFF library] and its documentation for any purpose is hereby granted without fee, provided that (i) the above copyright notices and this permission notice appear in all copies of the software and related documentation, and (ii) the names of Sam Leffler and Silicon Graphics may not be used in any advertising or publicity relating to the software without the specific, prior written permission of Sam Leffler and Silicon Graphics.

THE SOFTWARE IS PROVIDED “AS-IS” AND WITHOUT WARRANTY OF ANY KIND, EXPRESS, IMPLIED OR OTHERWISE, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

IN NO EVENT SHALL SAM LEFFLER OR SILICON GRAPHICS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT OR CONSEQUENTIAL DAMAGES OF ANY KIND, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER OR NOT ADVISED OF THE POSSIBILITY OF DAMAGE, AND ON ANY THEORY OF LIABILITY, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

The following disclaimer is required by the Independent JPEG Group:

This software is based in part on the work of the Independent JPEG Group.

This Software may contain components licensed under the following conditions:

Copyright (c) 1989 The Regents of the University of California. All rights reserved.

Redistribution and use in source and binary forms are permitted provided that the above copyright notice and this paragraph are duplicated in all such forms and that any documentation, advertising materials, and other materials related to such distribution and use acknowledge that the software was developed by the University of California, Berkeley. The name of the University may not be used to endorse or promote products derived from this software without specific prior written permission. THIS SOFTWARE IS PROVIDED ``AS IS'' AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Copyright (C) 1989, 1991 by Jef Poskanzer.

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation. This software is provided "as is" without express or implied warranty.

Copyright 1995, Trinity College Computing Center. Written by David Chappell.

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation. This software is provided "as is" without express or implied warranty.

Copyright 1996 Daniel Dardailler.

Permission to use, copy, modify, distribute, and sell this software for any purpose is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of Daniel Dardailler not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. Daniel Dardailler makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

Modifications Copyright 1999 Matt Koss, under the same license as above.

Copyright (c) 1991 by AT&T.

Permission to use, copy, modify, and distribute this software for any purpose without fee is hereby granted, provided that this entire notice is included in all copies of any software which is or includes a copy or modification of this software and in all copies of the supporting documentation for such software.

The following disclaimer is required by Paradigm Matrix:

Portions of this software licensed from Paradigm Matrix.

The following disclaimer is required by Ray Sauers Associates, Inc.:

“Install-It” is licensed from Ray Sauers Associates, Inc. End-User is prohibited from taking any action to derive a source code equivalent of “Install-It,” including by reverse assembly or reverse compilation, Ray Sauers Associates, Inc. shall in no event be liable for any damages resulting from reseller’s failure to perform reseller’s obligation; or any damages arising from use or operation of reseller’s products or the software; or any other damages, including but not limited to, incidental, direct, indirect, special or consequential Damages including lost profits, or damages resulting from loss of use or inability to use reseller’s products or the software for any reason including copyright or patent infringement, or lost data, even if Ray Sauers Associates has been advised, knew or should have known of the possibility of such damages.

The following disclaimer is required by Videomedia, Inc.:

“Videomedia, Inc. makes no warranties whatsoever, either express or implied, regarding this product, including warranties with respect to its merchantability or its fitness for any particular purpose.”

“This software contains V-LAN ver. 3.0 Command Protocols which communicate with V-LAN ver. 3.0 products developed by Videomedia, Inc. and V-LAN ver. 3.0 compatible products developed by third parties under license from Videomedia, Inc. Use of this software will allow “frame accurate” editing control of applicable videotape recorder decks, videodisc recorders/players and the like.”

The following disclaimer is required by Altura Software, Inc. for the use of its Mac2Win software and Sample Source Code:

The following disclaimer is required by 3Prong.com Inc.:

Certain waveform and vector monitoring capabilities are provided under a license from 3Prong.com Inc.

The following disclaimer is required by Interplay Entertainment Corp.:

The “Interplay” name is used with the permission of Interplay Entertainment Corp., which bears no responsibility for Avid products.

This product includes portions of the Alloy Look & Feel software from Incors GmbH.

This product includes software developed by the Apache Software Foundation (http://www.apache.org/).

This product may include the JCifs library, for which the following notice applies:

Avid Interplay contains components licensed from LavanTech. These components may only be used as part of and in connection with Avid Interplay.

Attn. Government User(s). Restricted Rights Legend

U.S. GOVERNMENT RESTRICTED RIGHTS. This Software and its documentation are “commercial computer software” or “commercial computer software documentation.” In the event that such Software or documentation is acquired by or on behalf of a unit or agency of the U.S. Government, all rights with respect to this Software and documentation are subject to the terms of the License Agreement, pursuant to FAR §12.212(a) and/or DFARS §227.7202-1(a), as applicable.

Trademarks

Avid, the Avid Logo, Avid Everywhere, Avid DNXHD, Avid DNXHR, Avid Nexis, AirSpeed, Eleven, EUCON, Interplay, iNEWS, ISIS, Mbox, MediaCentral, Media Composer, NewsCutter, Pro Tools, ProSet and RealSet, Maestro, PlayMaker, Sibelius, Symphony, and all related product names and logos, are registered or unregistered trademarks of Avid Technology, Inc. in the United States and/or other countries. The Interplay name is used with the permission of the Interplay Entertainment Corp. which bears no responsibility for Avid products. All other trademarks are the property of their respective owners. For a full list of Avid trademarks, see: http://www.avid.com/US/about-avid/legal-notices/trademarks.

Footage

WCAU Fire Story — Courtesy of NBC-10, Philadelphia, PA.

News material provided by WFTV Television Inc.

Avid MediaCentral Platform Services — Concepts and Clustering Guide • Created 6/23/17 • This document is distributed by Avid in online (electronic) form only, and is not available for purchase in printed form.

Contents

Using This Guide

This guide is intended for individuals responsible for installing, maintaining or performing administrative tasks on an Avid MediaCentral Platform Services (MCS) system. This document serves as an educational tool; providing background and technical information on MCS. Additionally, it explains the specifics of an MCS cluster, how each service operates in a cluster, and provides guidance on best practices for the administration of an MCS system.

In addition to this guide, reference the following companion documents for more information:

- *Avid MediaCentral Platform Services ReadMe* – Read prior to completing any MCS installation or upgrade.
- *Avid MediaCentral Platform Services Installation and Configuration Guide* – Reference if you are installing MCS on a new server or cluster of servers.
- *Avid MediaCentral Platform Services Upgrade Guide* – Reference if you are upgrading MCS from a previous release.
- *Avid MediaCentral | UX Administration Guide* – Contains administrative information for MediaCentral UX.
- *Avid Media | Index Configuration Guide* – Reference if configuring Media Index.
- *Avid MediaCentral Platform Services Hardware Guide* - Provides detailed information on HP and Dell servers.
- *Avid MediaCentral Platform Services Virtual Environment with VMware® Best Practices Guide* - Provides detailed information on configuring a virtual MCS environment with VMware.

Important: See the following link on the Avid Knowledge Base for the latest updates to this guide and all related documentation:

http://avid.force.com/pkb/articles/en_US/user_guide/Avid-MediaCentral-Documentation

Important Terms

Throughout this document, “Avid MediaCentral Platform Services” is referred to as “MCS”. “Red Hat Enterprise Linux” is referred to as “RHEL”.

The RHEL deployment used in an MCS environment is a command-line based operating system. The installation process requires the editing of various system files. Although multiple text editors exist, the tool used throughout this document is “vi”. For a short introduction to vi, see “Working with Linux” on page 16.

⚠ **If copying / pasting commands from this document into a command line interface such as Putty, be sure to verify the command after pasting. It is possible that some characters might be replaced during the paste process which can lead to a failed or problematic installation.**

When working in Linux, this guide assumes the user is logged in as the “root” user. Perform all commands and server configuration as the “root” user.

Symbols and Conventions

Avid documentation uses the following symbols and conventions:

Symbol or Convention	Meaning or Action
🗎	A note provides important related information, reminders, recommendations, and strong suggestions.
⚠	A caution means that a specific action you take could cause harm to your computer or cause you to lose data.
⚡	A warning describes an action that could cause you physical harm. Follow the guidelines in this document or on the unit itself when handling electrical equipment.
>	This symbol indicates menu commands (and subcommands) in the order you select them. For example, File > Import means to open the File menu and then select the Import command.
▶	This symbol indicates a single-step procedure. Multiple arrows in a list indicate that you perform one of the actions listed.
(Windows), (Windows only), (Macintosh), or (Macintosh only)	This text indicates that the information applies only to the specified operating system, either Windows or Macintosh OS X.
Bold font	Bold font is primarily used in task instructions to identify user interface items and keyboard sequences.
Italic font	Italic font is used to emphasize certain words and to indicate variables.
`Courier Bold font`	Courier Bold font identifies text that you type.
Ctrl+key or mouse action	Press and hold the first key while you press the last key or perform the mouse action. For example, Command+Option+C or Ctrl+drag.

If You Need Help

If you are having trouble using your Avid product:

1. Retry the action, carefully following the instructions given for that task in this guide. It is especially important to check each step of your workflow.
2. Check the latest information that might have become available after the documentation was published. You should always check online for the most up-to-date release notes or ReadMe because the online version is updated whenever new information becomes available. To view these online versions, select ReadMe from the Help menu, or visit the Avid MediaCentral Platform Services documentation page at: http://avid.force.com/pkb/articles/en_US/user_guide/Avid-MediaCentral-Documentation
3. Check the documentation that came with your Avid application or your hardware for maintenance or hardware-related issues.
4. Visit the Avid Knowledge Base. Online services are available 24 hours per day, 7 days per week. Search this online Knowledge Base to find answers, to view error messages, to access troubleshooting tips, to download updates, and to read or join online message-board discussions.

Avid Training Services

Avid makes lifelong learning, career advancement, and personal development easy and convenient. Avid understands that the knowledge you need to differentiate yourself is always changing, and Avid continually updates course content and offers new training delivery methods that accommodate your pressured and competitive work environment.

For information on courses/schedules, training centers, certifications, courseware, and books, please visit www.avid.com/support and follow the Training links, or call Avid Sales at 800-949-AVID (800-949-2843).

1 Overview

Avid MediaCentral Platform Services (MCS) is a collection of services running on one or more servers, providing a base infrastructure for solutions including MediaCentral UX, Media Composer Cloud, and Interplay MAM. Multiple MCS servers can be grouped together in a cluster configuration to provide high-availability and increased scale. Every server in a cluster is identified as a “node”. The first two nodes in a cluster are known as the primary (master) and secondary (slave). Any additional server in the cluster is known as a load-balancing node.

In a single-server configuration, all MCS services run on the single, standalone node. In a cluster, the primary and secondary nodes run all services; while a limited number of services run on the load-balancing nodes. Select services on the secondary cluster node wait in standby and only become active in the event of a failure of the primary node. If a failure occurs, the services automatically start on the secondary node, without the need for human intervention which greatly reduces system downtime.

When increased client and stream-counts are required, load-balancing servers can be added to the cluster. Load-balancing nodes add scale to the system, but they do not participate in failover. If both the primary and secondary nodes are offline, the MCS system will be down until one of these servers becomes available. A load-balanced cluster provides better performance for deployments supporting multiple, simultaneous users or connections.

An additional benefit of a load-balanced cluster is *cache replication*, in which media transcoded by one server is immediately distributed to all the other nodes in the cluster. If node-2 receives a playback request for media already transcoded by node-1, the material is immediately available on node-2 without the need for re-transcoding. Cache replication is achieved through an open source, distributed file system called GlusterFS.

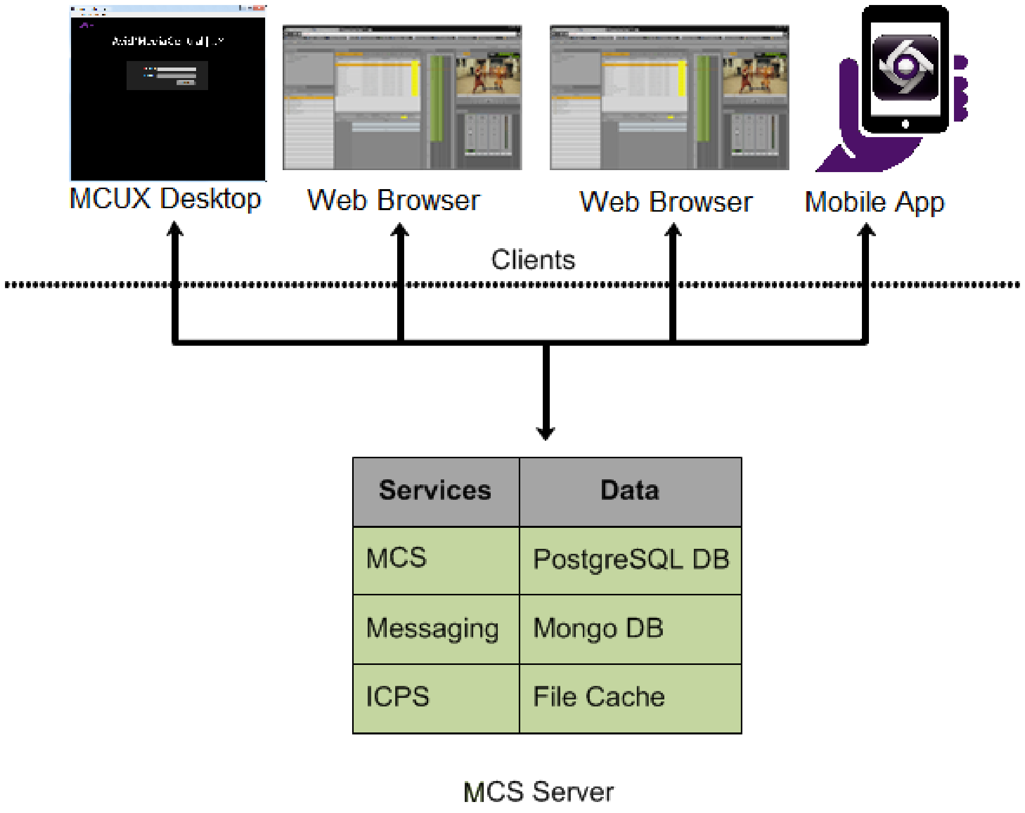
In summary, an MCS cluster differentiates itself from a single server by providing the following:

- **Redundancy/High-Availability**. Services are mirrored on the primary and secondary nodes which provide redundancy of databases, system settings and key services. If any node in the cluster fails, connections to that node are automatically redirected to another node.
- **Scale/Load-Balancing**. Multiple servers can be added to the cluster to increase the total possible number of client connections and playback streams. All incoming playback connections are routed to a single cluster IP address, and are subsequently distributed evenly across the nodes in the cluster.
- **Replicated Cache**. The media transcoded by one node in the cluster is automatically replicated on the other nodes. If “node2” receives a playback request for an asset that was already transcoded on “node1”, the replication process ensures that the media is available on “node2” without the need to re-transcode.
- **Cluster Monitoring**. The clustering software includes a utility that enables system administrators to monitor the status of all cluster resources and related services on all nodes. In addition, if a node fails or if a serious problem is detected, designated users are alerted to the issue through an automatically generated an e-mail.

Single Server Deployments

In a single server deployment, all MCS services (including the playback service) run on the same server. This server also hosts the MCS database and a file cache which contains the transcoded media files used in playback requests. The MCS server is assigned a standard host name and IP address. MediaCentral UX users connect directly to this server through access portals such as the MediaCentral UX Desktop application, designated web browsers, or the MediaCentral UX mobile app.

The following diagram illustrates a typical single-server deployment:



Services	Data
MCS	PostgreSQL DB
Messaging	Mongo DB
ICPS	File Cache

For a list of supported web browsers, see the Interplay Production and MediaCentral Compatibility Matrix on the Avid Knowledge Base.

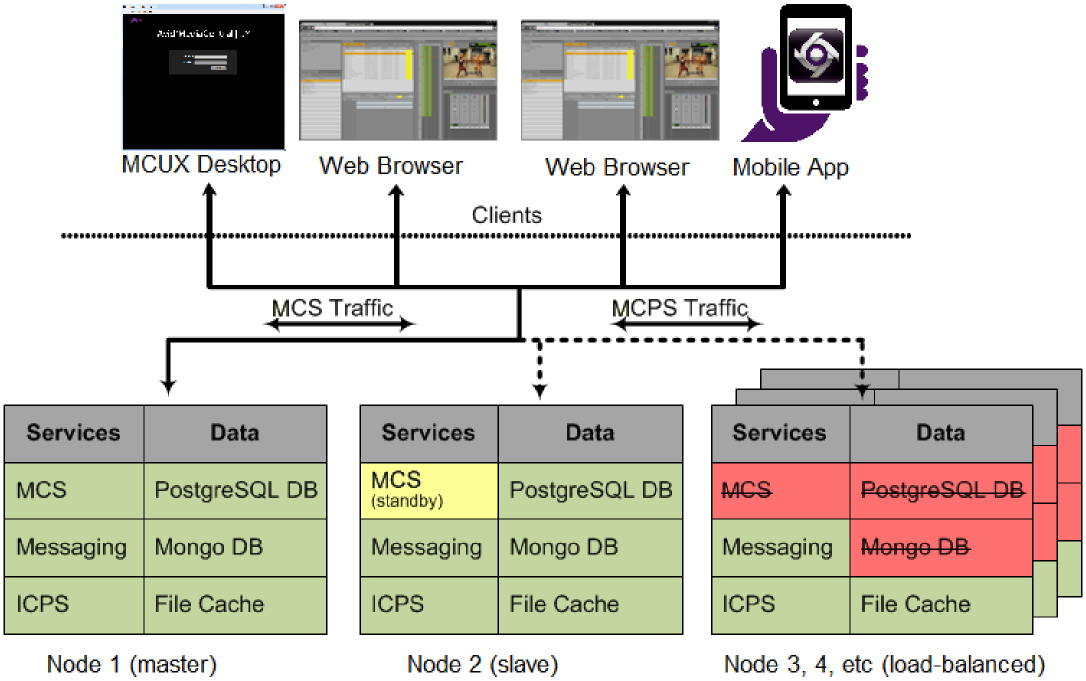
Multi-Server Deployments

Two or more MCS servers connect to each other through clustering software installed and configured on each server. In a basic deployment, a cluster consists of a master/slave pair of nodes configured for high-availability. All MCS traffic is routed through the master node which is running all MCS services. Select MCS services and databases are replicated to the slave node. Some of these services are actively running while others are in “standby” mode; ready to assume the role of master at any time. Although not required, additional nodes are often present in a cluster configuration to support load-balanced transcoding, playback and increased scale.

Playback requests, handled by the ICPS playback service, are received by the master and distributed to available nodes. The load-balancing nodes perform transcoding and playback, but do not participate in failover. Unless reconfigured by a system administrator, the load-balancing nodes can never take on the role of master or slave.

An interesting difference in a cluster deployment is at the network level. In a single server deployment, the MCS server owns its host name and IP address. Clients connect directly to this server to access the MCS system. In a cluster configuration, while each server maintains its own host name and IP address, a virtual host name and IP address is also configured for the cluster group. MediaCentral UX users connect to the cluster’s virtual host name (FQDN), and not to the name of an individual server. Connecting to the cluster and not to an individual node ensures that the client request is always serviced, regardless of which nodes may be available at the time.

The following diagram illustrates a typical cluster deployment:



Services	Data
MCS	PostgreSQL DB
Messaging	Mongo DB
ICPS	File Cache

Services	Data
MCS (standby)	PostgreSQL DB
Messaging	Mongo DB
ICPS	File Cache

Services	Data
~~MCS~~	~~PostgreSQL DB~~
Messaging	~~Mongo DB~~
ICPS	File Cache

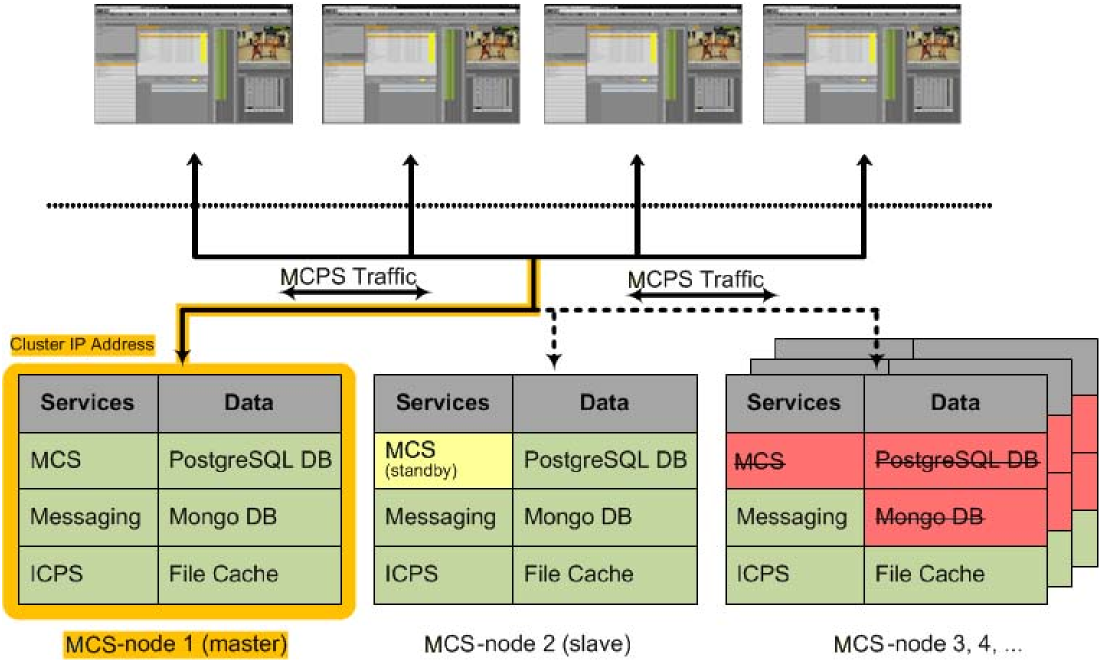
- Services / systems in green are active on that node.
- Services / systems in yellow are in a standby mode.
- Services / systems in red are disabled on that node.

How Failover Works

Failover in MCS operates at two distinct levels: service, and node - both of which are manged by a cluster monitoring system. If a service fails, it is quickly restarted by the cluster manager, which also tracks the service's fail-count. If the service fails too often (or cannot be restarted), the cluster manager gives responsibility for the service to the standby node in the cluster, in a process referred to as a *failover*. A service restart in itself is not enough to trigger a failover. A failover occurs when the fail-count for the service reaches a specified threshold value.

The node on which the service failed remains in the cluster, but no longer performs the duties that have failed. Until the fail-count is manually reset, the failed service will not be restarted.

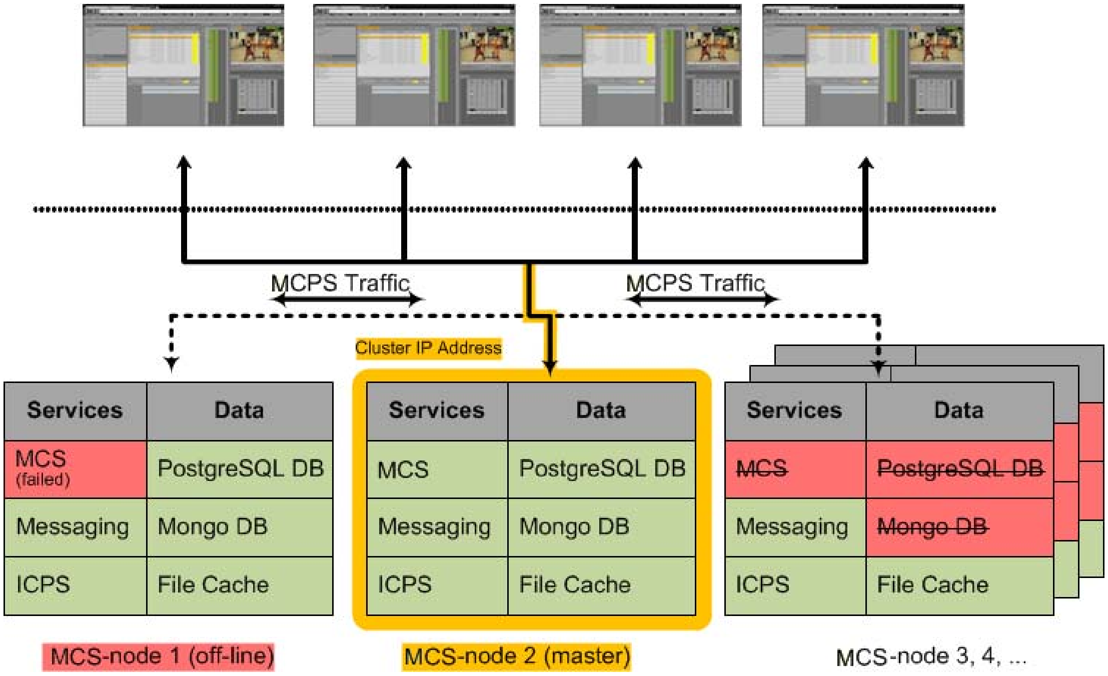
In order to achieve this state of high-availability, one node in the cluster is assigned the role of master. It runs all the key MCS services. The master node also owns the cluster IP address. Thus all incoming requests come directly to this node and are serviced by it. This is shown in the following illustration:



Should any of the key MCS services running on the master node fail without recovery (or reach the failure threshold) a failover is initiated and the secondary node takes on the role of master node. The node that becomes master inherits the cluster IP address, and its own MCS services (that were previously in standby) become fully active. From this point, the new master receives all incoming requests. Administrators must manually intervene to determine the cause of the fault on the failed node and to restore it to service.

In a correctly sized cluster, a single node can fail and the cluster will properly service its users. However, if two nodes fail, the remaining servers are likely under-provisioned for expected use and will be oversubscribed. Users should expect reduced performance in this scenario. If the primary and secondary nodes both fail, the system will be unavailable until the situation is resolved.

The failover from master to slave is shown in the following illustration:



MCS-node 1 (off-line)

Services	Data
MCS (failed)	PostgreSQL DB
Messaging	Mongo DB
ICPS	File Cache

MCS-node 2 (master)

Services	Data
MCS	PostgreSQL DB
Messaging	Mongo DB
ICPS	File Cache

MCS-node 3, 4, ...

Services	Data
~~MCS~~	~~PostgreSQL DB~~
Messaging	~~Mongo DB~~
ICPS	File Cache

How Load-Balancing Works

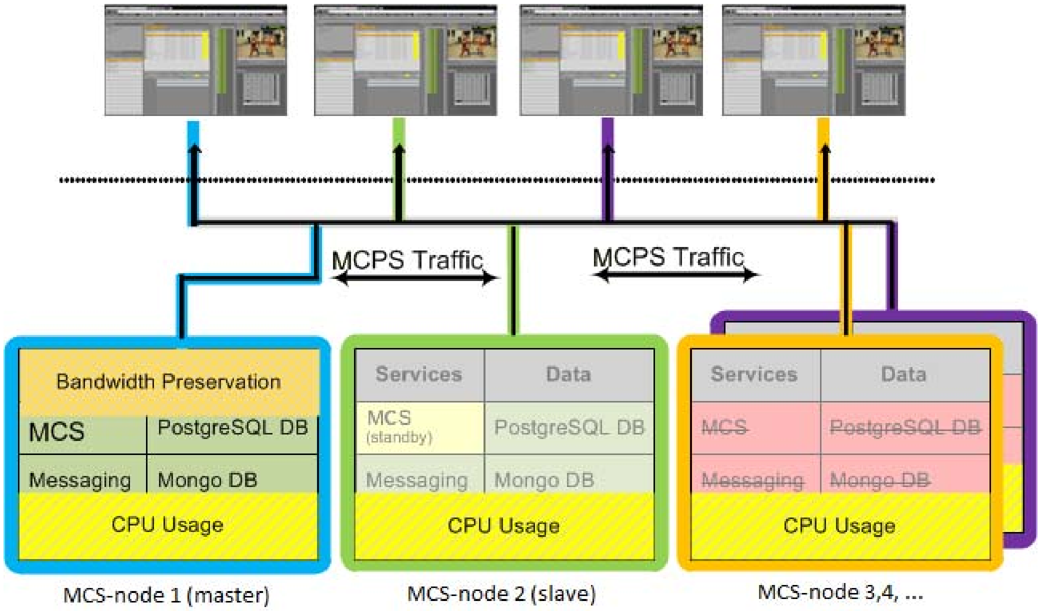
MCS video playback is load-balanced, meaning that incoming video playback requests are distributed across all nodes in the cluster. Playback is made possible through the Interplay Central Playback Service (ICPS) which actively runs on all nodes in the cluster concurrently.

When a client generates a playback request, the task is received by the master node. A load-balancing algorithm controlled by the master node monitors the clustered nodes, and distributes the request to a playback node. The playback node reads the source media from a shared storage system and performs a quick lower-resolution transcode to stream to the client.

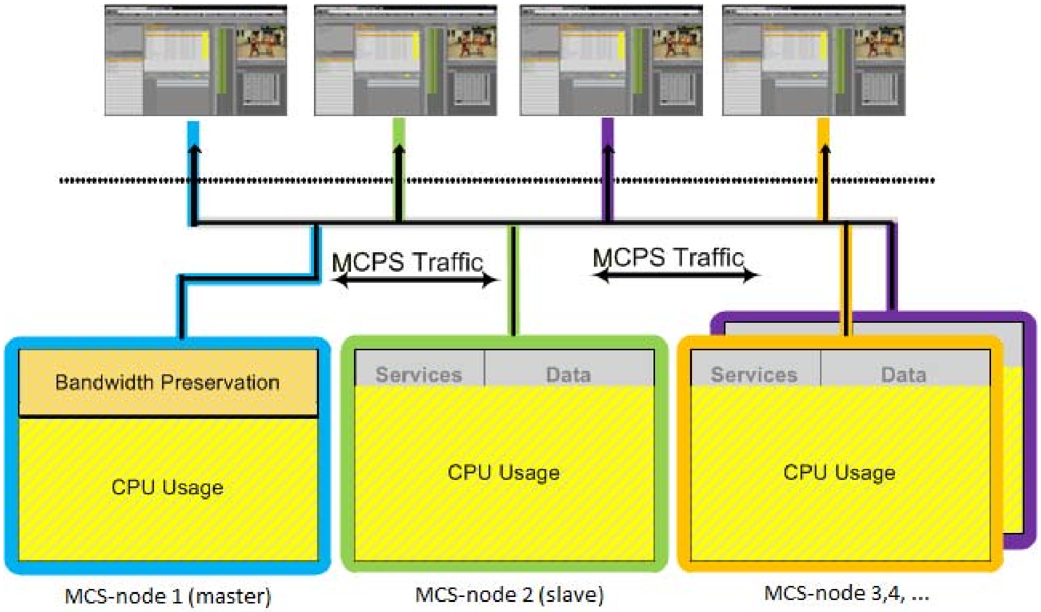
The node that has the least amount of system load receives the playback request. Subsequent playback requests continue in a “round-robin” style where the next most available node receives the following playback request.

The master node is treated differently in that 30% of its CPU capacity is always reserved for the duties performed by the master node alone, which include serving the UI, handling logins and user session information, and so on. When the system is under heavy usage, the master node will not take on additional playback jobs. All other nodes can reach 100% CPU saturation to service playback requests.

The following illustration shows a typical load-balanced cluster. The colored lines indicate that playback jobs are sent to different nodes in the cluster. They are not meant to indicate a particular client is bound to a particular node for its entire session, which may not be the case. Notice the master node's bandwidth preservation.



The next illustration shows a cluster under heavy usage. As illustrated, CPU usage on the master node will not exceed a certain amount, even when the other nodes approach saturation.



Working with Linux

Red Hat Enterprise Linux (RHEL) is a commercially supported, open source version of the Linux operating system. If you have run DOS commands in Windows or have used the Mac terminal window, the Linux environment will be familiar to you. While many aspects of the MCS installation are automated, much of the configuration process requires entering commands and editing files using the Linux command-line.

Red Hat Enterprise Linux is not free, and Avid does not redistribute it or include it as part of the MCS software package. RHEL licensing and support options are covered in the MediaCentral Platform Services Hardware Guide.

Installing Linux

Installations on Avid qualified HP and Dell servers use an express process involving a USB drive and an Avid-supplied "kickstart" file (ks.cfg). Kickstart files are commonly used in Linux deployments to automate the software installation by automatically answering questions posed by the Linux installer, for hardware known in advance.

To further assist in the deployment, the MCS installation package includes a Windows-based tool called "ISO2USB". This application is used to create a bootable USB drive from a RHEL installation DVD or image (.iso) file. When a user boots from this USB drive, RHEL and the MCS software packages are installed simultaneously with limited involvement from the user.

If you are installing MediaCentral Platform Services on hardware that has not been qualified by Avid, see "Appendix A: Installing MCS on Non-HP / Dell Hardware" in the Avid MediaCentral Platform Services Installation and Configuration Guide.

Linux Concepts

Once RHEL is installed you can begin the work of configuring the server for MCS. This process includes simple actions such as verifying the system time as well as more complex actions, such as verifying and modifying a group of configuration files related to networking. Depending on the deployment, you may also be required to create logical volumes, configure port bonding, and perform other advanced actions.

Advance knowledge of the following Linux concepts is helpful:

- root user: The *root* user (sometimes called the "super" user) is the Linux user with highest privileges. All steps in the installation are performed as root.
- mounting: Linux does not recognize hard drives or removable devices such as USB keys unless they are formally mounted.
- file and directory structure: You will be required to navigate through the Linux directory structure without the assistance of a GUI (Graphical User Interface).

Accessing the Linux Command Line

The Linux command line interface is a powerful tool that lets you perform simple and complex actions alike with equal speed and ease. The command line can be accessed by connecting a keyboard directly to the server. However, it is often preferred to access the system indirectly over a network connection through an SSH (Secure Shell) utility such as PuTTY.

An SSH connection is preferable for the following reasons:

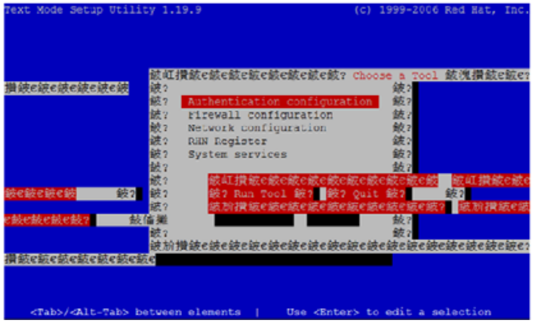
- Allows for an expandable view of the RHEL command line interface (adjustable window size)
- Allows for multiple sessions to the same host server or to multiple servers
- Allows for simplified copy/paste of commands between SSH windows (especially convenient when configuring a cluster)
- Allows for logging of all session output

Many SSH utilities can be found through a simple internet search, but PuTTY is the utility used in all examples throughout the Avid MediaCentral Platform Services documentation set.

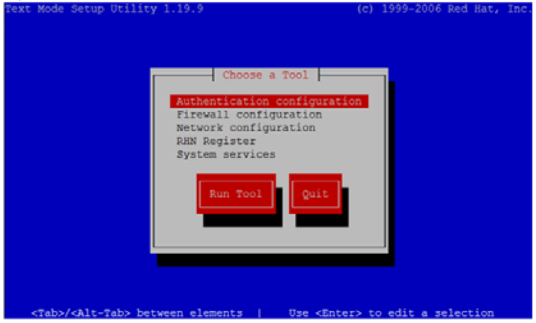
A Windows-based utility, PuTTY can be found and downloaded at the following location: http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html

Depending on your keyboard configuration, some languages such as Chinese or other multi-byte character sets might display the PuTTY output incorrectly. For example, the Linux `setup` command displays the graphical “Choose a Tool” menu which should include clean boarders:

Original Character set:


Text Mode Setup Utility 1.19.9
(c) 1999-2006 Red Hat, Inc.
Choose a Tool
Authentication configuration
Firewall configuration
Network configuration
RHN Register
System services
Run Tool
Quit
<Tab>/<Alt-Tab> between elements | Use <Enter> to edit a selection

Updated with the UTF-8 character set:


Text Mode Setup Utility 1.19.9
(c) 1999-2006 Red Hat, Inc.
Choose a Tool
Authentication configuration
Firewall configuration
Network configuration
RHN Register
System services
Run Tool
Quit
<Tab>/<Alt-Tab> between elements | Use <Enter> to edit a selection

If you are not seeing a clean image, you can adjust the “Remote character set” within PuTTY to alter the output. The setting is located in the PuTTY Configuration window under Window > Translation > Remote character set.

For more information on using PuTTY, see the documentation page at: http://www.chiark.greenend.org.uk/~sgtatham/putty/docs.html

Common Command Line Commands

There are a few simple commands that are used often throughout the initial configuration of the Linux server. For example, entering the Linux list command, *ls*, at the root (/) directory produces results similar to the following:

```
[root@localhost ~]# ls
bin    cgroup   Documents   lib    media  net   sbin     sys         usr  vol1
boot   Desktop  etc         lib64  misc   opt   selinux  sysinstall  var
cache  devhome  lost+found  mnt    proc   root  srv      tmp
```

Review the structure of the command used in the previous example and note the following:

- The pound sign (#) indicates the presence of the Linux command prompt for a user with root level privileges (the highest privilege level). You do not type a pound sign.

- A dollar sign ($) symbol (not shown) indicates that you are logged in as a standard user without root-level privileges.
- Linux commands, paths, and file names are case-sensitive.

The following table presents a few of the more commonly used Linux commands:

Command	**Description**
ls	Lists directory contents. Use the -l option (hyphen, lower-case L) for a detailed listing.
cd	Changes directories.
cat <*filename*>	Prints the contents of the named file to the screen.
clear	Clears screen.
cp	Copies files and directories.
<tab>	Auto-completes the command based on contents of the command line and directory contents. For example, typing cd and the beginning of a directory name, then pressing the tab key fills in the remaining letters in the name.
|	"Pipes" the output from one command to the input of another. For example, to view the output of a command one screen at a time, pipe into the more command, as in: ls | more
dmesg	Displays messages from the Linux kernel buffer. Useful to see if a device (such as USB key) mounted correctly.
find	Searches for files. For example, the following use of the find command searches for <*filename*> on all local filesystems (avoiding network mounts): find / -mount -name <*filename*> If you can not find the file name, try using a wild card (*) at the beginning or end of the filename.
grep	Searches for the named regular expression. Often used in conjunction with the pipe command, as in: `ps | grep avid` This example would display all running processes that contain the word "avid".
less	Similar to the cat command, but automatically breaks up the output in to screen-sized chunks, with navigation. Useful for navigating large amounts of text on screen at a time. For example: `less <filename>`
locate	The locate command works much like the `find` command to locate files on the system. For example: `locate <filename>` If you can not find the file name, try using a wild card (*) at the beginning or end of the file name.
lvdisplay	Displays information about logical volumes.

Command	Description
man	Presents help (the “manual page”) for the named command.
mkdir	Creates a new directory.
| more	Piping (“|”) the output of a command through the more command breaks up the output into screen-sized chunks. For example to view the contents of a large directory one screen at a time, type the following: `ls | more`
mount umount	Mounts or unmounts an external device to a directory. A device must be mounted before its contents can be accessed.
ps	Lists the running processes.
passwd	Changes the password for the logged-in user.
scp	Securely copies files between machines (across an ssh connection).
tail	Shows you the last 10 (or n) lines in a file. tail <*filename*> tail -50 <*filename*> tail –f <*filename*> The “-f” option keeps the tail command outputting appended data as the file grows. Useful for monitoring log files.
udevadm	Requests device events from the Linux kernel. Can be used to replay device events and create/update the 70-persistent-net.rules file. e.g. udevadm trigger --action=add
vi	Starts a vi editing session.

Key Linux Directories

Like other file systems, the Linux filesystem is represented as a hierarchical tree. In Linux directories are reserved for particular purposes. The following table presents some important Linux directories encountered during the MCS installation and configuration:

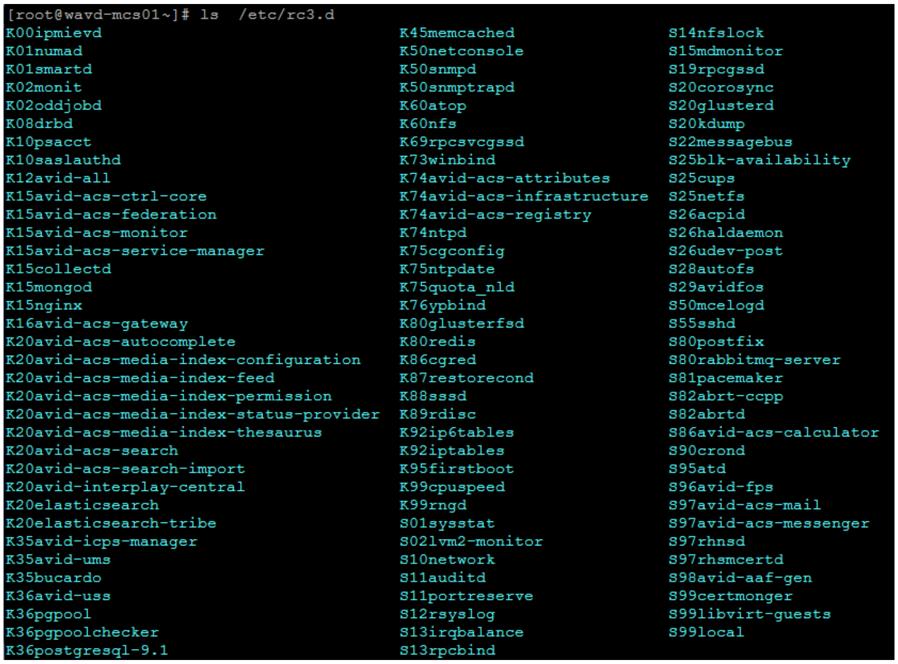
Directory	Description
/	The root of the filesystem.
/dev	Contains device files, including those identifying HD partitions, USB and CD drives, and so on. For example, sda1 represents the first partition (1) of the first hard disk (a).
/etc	Contains Linux system configuration files, including the filesystem table, fstab, which tells the operating system what volumes to mount at boot-time.
/etc/udev/rules.d	Contains rules used by the Linux device manager, including network script files where persistent names are assigned to network interfaces. In Linux, every network interface has a unique name. For example, if a server has four network ports, they might be named *eth0* through *eth3*.

Directory	Description
/etc/sysconfig/network-scripts	Contains, amongst other things, files providing Linux with boot-time network configuration information, including which network interfaces to bring up.
/media	Contains the mount points for detachable storage, such as USB keys. In Linux, volumes and removable storage must be mounted before they can be accessed.
/opt	Contains add-on application packages that are not a native part of Linux, including the MCS components.
/usr	Contains user binaries, including some MCS components.
/tmp	The directory for temporary files.
/var	Contains data files that change in size (variable data), including the MCS server log files.

Linux Run Levels

Avid MediaCentral Platform Services uses Linux Run Level 3, which is RHEL with no graphical user interface (GUI). The start order assigned to Linux services is defined per the **/etc/rc3.d** directory. The files in this directory are prefixed **Sxx** or **Kxx** (e.g. S24, S26, K02). The **Sxx** prefix indicates that the service or daemon is started at boot time. The **Kxx** prefix indicates that the service or daemon is killed at boot time. The naming conversion ensures that items are killed and started in a specific order.

The following is an example output of the **/etc/rc3.d** directory:



```
[root@wavd-mcs01~]# ls  /etc/rc3.d
K00ipmievd                               K45memcached                S14nfslock
K01numad                                 K50netconsole               S15mdmonitor
K01smartd                                K50snmpd                    S19rpcgssd
K02monit                                 K50snmptrapd                S20corosync
K02oddjobd                               K60atop                     S20glusterd
K08drbd                                  K60nfs                      S20kdump
K10psacct                                K69rpcsvcgssd               S22messagebus
K10saslauthd                             K73winbind                  S25blk-availability
K12avid-all                              K74avid-acs-attributes      S25cups
K15avid-acs-ctrl-core                    K74avid-acs-infrastructure  S25netfs
K15avid-acs-federation                   K74avid-acs-registry        S26acpid
K15avid-acs-monitor                      K74ntpd                     S26haldaemon
K15avid-acs-service-manager              K75cgconfig                 S26udev-post
K15collectd                              K75ntpdate                  S28autofs
K15mongod                                K75quota_nld                S29avidfos
K15nginx                                 K76ypbind                   S50mcelogd
K16avid-acs-gateway                      K80glusterfsd               S55sshd
K20avid-acs-autocomplete                 K80redis                    S80postfix
K20avid-acs-media-index-configuration    K86cgred                    S80rabbitmq-server
K20avid-acs-media-index-feed             K87restorecond              S81pacemaker
K20avid-acs-media-index-permission       K88sssd                     S82abrt-ccpp
K20avid-acs-media-index-status-provider  K89rdisc                    S82abrtd
K20avid-acs-media-index-thesaurus        K92ip6tables                S86avid-acs-calculator
K20avid-acs-search                       K92iptables                 S90crond
K20avid-acs-search-import                K95firstboot                S95atd
K20avid-interplay-central                K99cpuspeed                 S96avid-fps
K20elasticsearch                         K99rngd                     S97avid-acs-mail
K20elasticsearch-tribe                   S01sysstat                  S97avid-acs-messenger
K35avid-icps-manager                     S02lvm2-monitor             S97rhnsd
K35avid-ums                              S10network                  S97rhsmcertd
K35bucardo                               S11auditd                   S98avid-aaf-gen
K36avid-uss                              S11portreserve              S99certmonger
K36pgpool                                S12rsyslog                  S99libvirt-guests
K36pgpoolchecker                         S13irqbalance               S99local
K36postgresql-9.1                        S13rpcbind
```

The Linux start order as reflected in the ***/etc/rc3.d*** *and the other run-level directories ("/etc/rc****x****.d") reflect the boot order for the server. The shutdown order is reflected in the* ***/etc/rc0.d*** *directory. They do not always reflect dependencies within MCS itself.*

Linux Text Editor (vi)

Linux features a powerful text editor called vi. To invoke the command, type “vi”, followed by the target file. If you are not in the directory containing the file to be edited, you must also enter a file path.

```
vi [path]/<filename>
```

vi operates in one of two modes, insert mode and command mode. Insert mode lets you perform text edits – insertion, deletion, etc. Command mode acts upon the file as a whole – for example, to save it or to quit without saving.

- Press the “i” (as in Indigo) key to switch to insert mode.
- Press the colon (“:”) key to switch to command mode.

The following table presents a few of the more useful vi command mode commands:

Key Press	Description
:	Prefix to commands in command mode
:wq	Write file and quit vi (in command mode)
:q!	Quit without writing (in command mode)

The following table presents a few of the more useful vi insert mode commands:

Key Press	Description
i	Insert text before the cursor, until you press <Esc>
I	Insert text at beginning of current line
a	Insert text after the cursor
A	Insert text at end of current line
w	Next word
b	Previous word
Shift-g	Move cursor to last line of the file
D	Delete remainder of line
x	Delete character under the cursor
dd	Delete current line
yy	“Yank” (copy) a whole line in command mode.
p	Paste the yanked line in command mode.
<Esc>	Turn off Insert mode and switch to command mode.

For more information on vi commands, see the following link:

https://www.cs.colostate.edu/helpdocs/vi.html

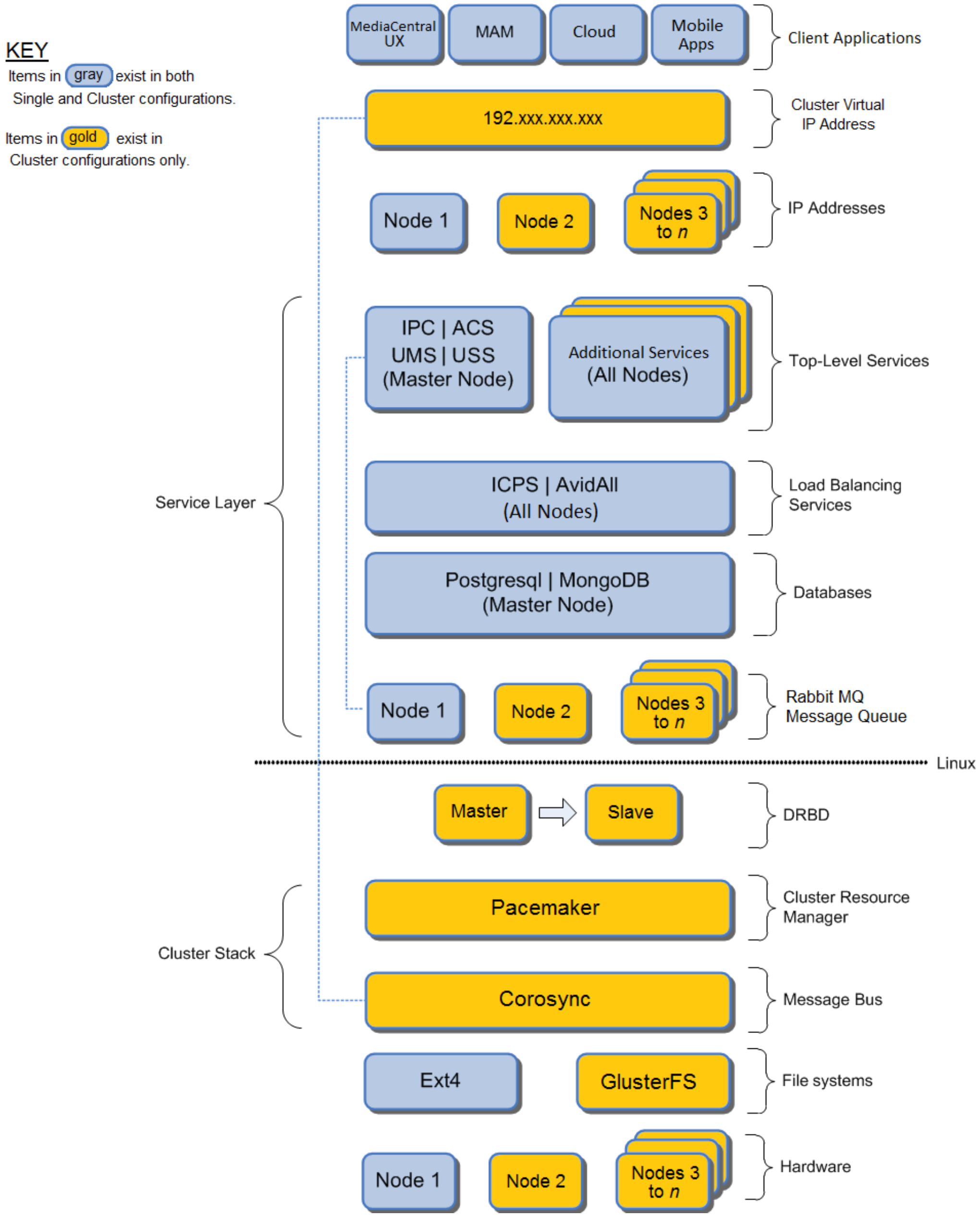
Linux Usage Tips

The following table presents tips that will make it easier to work in RHEL:

Tip	Description
Using the Tab key	Your keyboard’s Tab key can be used to auto-complete directory paths and file names. For example, if you want to navigate to the `/var/log/avid/media-index` directory type: **`cd /var/log/avid/m`** ...and then press the Tab key. The remaining path is completed for you. If the path contains two folders that contain part of the same name, quickly press tab twice to show a list of all files and folders with that name. For example, type: **`cd /var/log/avid/a`** ...and quickly press the Tab key twice. The tab key can also be used with other commands to complete long file names. For example, if you are unzipping the Avid MediaCentral Closed Captioning service prior to installation, pressing the Tab key after typing: **`unzip M`** ...results in the completed file name: `unzip MediaCentral_ClosedCaptioning_Service_2.8_Linux.zip` Since all commands and file names in RHEL are case-sensitive, using the tab key not only accelerates command entry, but also increases the accuracy of the commands.
Getting Help	For help with Linux commands, the Linux System Manual (“man” pages) are easily available by typing the man command followed by the item of interest. For example, for help with the ls command, type: `man ls`
Searching within a *man* page	To search for a string within a Linux *man* page, type the forward slash (“/”) followed by the string of interest. This can be helpful for finding a parameter of interest in a long *man* entry.
“command not found” error	A common experience for users new to the Linux command line is to receive a “command not found” after invoking a command or script that is definitely in the current directory. Linux has a PATH variable, but for reasons of security, the current directory — “.” in Linux — is not included in it by default. Thus, to execute a command or script in a directory that is unknown to the PATH variable you must enter the full path to the script from the root directory (“/”) or if you have already navigated to the desired directory, using the dot-slash (“./”) notation tells Linux that you are looking for a file in the current directory.
Copy and Paste with PuTTY	Using a mouse to highlight text in PuTTY automatically copies the text into the Windows Clipboard. Right-clicking in a second PuTTY window pastes the information from the Clipboard.

2 System Architecture

Avid MediaCentral Platform Services is comprised of multiple systems such as: messaging systems, user management services, cluster management infrastructure, and so on. While many of these systems are independent, they are required to work together to create a cohesive environment. The following diagram shows how these systems operate at distinct layers of the architecture.



The following table explains the role of each layer:

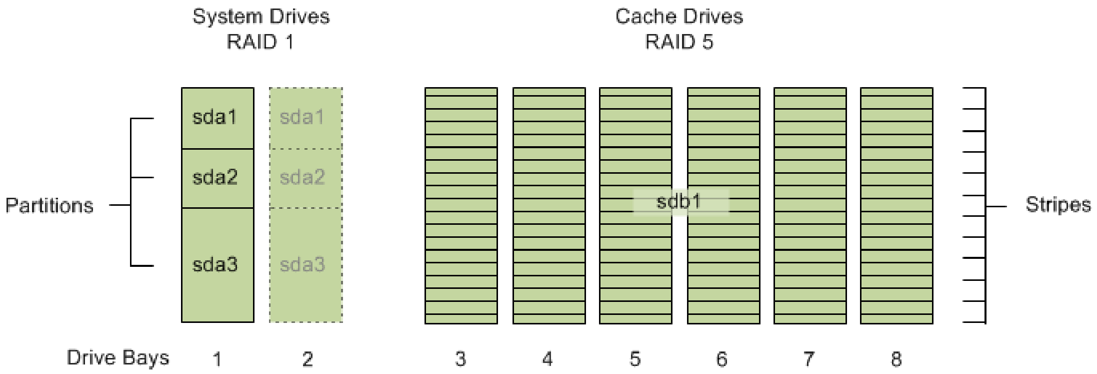
System Architecture Layer	Description
Client Applications	MCS clients are defined as any system that takes advantage of the MCS platform. Clients can range in complexity from a single MediaCentral UX session on a web browser to a complex system such as Interplay MAM. Additional client examples include Media Composer Cloud, and MediaCentral UX on a mobile device.
Cluster Virtual IP Address	In a cluster, clients gain access to MCS via the cluster's virtual IP address. The dotted line in the illustration indicates that Corosync manages ownership of the Cluster IP address.
Node IP Addresses	In a single server configuration, a standard unicast IP address is assigned to the server. In a cluster configuration, each server is assigned its own IP address and host name. However, the cluster is seen from the outside as a single (virtual) server with its own IP address and host name.
Top-Level Services	At the top level of the service layer are the MCS services running on a single server or cluster master node only. These include: • IPC - Interplay Central core services (aka "middleware") • UMS - User Management Services • USS - User Setting Service • ACS - Avid Common Service bus (aka "the bus") (configuration & messaging uses RabbitMQ. The dotted line in the illustration indicates the top level services communicate with one another via ACS, which, in turn, uses RabbitMQ. Additional Services - These services might not be active on all systems as they require additional software or configuration. • Media Distribute services • Media Index services • Closed Captioning service
Load-Balancing Services	The mid-level service layer includes the services that run on all servers, regardless of a single server or cluster configuration. In a cluster, these services are load-balanced. • AvidConnectivityMon - Verifies that the "always on" cluster IP is reachable. • AvidAll - Encapsulates all other ICPS back-end services. • AvidICPS - Interplay Central Playback Services: Transcodes and serves transcoded media for playback.

System Architecture Layer	Description
Databases	The mid-level service layer also includes two databases: • PostgreSQL: Stores data for several MCS services (UMS, ACS, ICPS). • MongoDB: Stores data related to MCS messaging. • Sharded MongoDB: Back-end for services such as avid-iam and avid-asset In a cluster configuration, these databases are synchronized between the master and slave nodes for failover readiness.
RabbitMQ Message Queue	RabbitMQ is the message broker (“task queue”) used by the MCS top level services. In a cluster, RabbitMQ maintains its own independent clustering system. That is, RabbitMQ is not managed by Pacemaker. This allows RabbitMQ to continue delivering service requests to underlying services in the event of a failure.
DRBD	Distributed Replicated Block Device (DRBD) is responsible for volume mirroring. DRBD replicates and synchronizes the system disk's logical volume containing the PostgreSQL and MongoDB databases across the master and slave, for failover readiness. DRBD carries out replication at the block level.
Pacemaker	The cluster resource manager. A resource represents a service or a group of services that are monitored by Pacemaker. Pacemakers sees and manages resources, not individual services.
Corosync	Corosync is the clustering infrastructure. By default, Corosync uses a multicast address to communicate with the other nodes in the cluster. However, configurations can be modified to use unicast addresses for networks that do not support multicast protocols.
File systems	The standard Linux file system. This layer also conceptually includes GlusterFS, the Gluster “network file system” used for cache replication. GlusterFS performs its replication at the file level. Unlike the Linux file system, GlusterFS operates in the “user space” - the advantage being any GlusterFS malfunction does not bring down the system.
Hardware	At the lowest layer is the server hardware. This includes network adapters, disk drives, BIOS settings and more. The system disk is established in a RAID 1 (mirrored) configuration. This mirroring is distinct from the replication of a particular volume by DRBD in a cluster configuration. The RAID 1 mirror protects against disk failure. The DRBD mirror protects against node failure. Many systems also include multiple disks in a RAID 5 configuration. These disks are configured as a cache for the low resolution transcoded media that is streamed to the clients.

The following sections of this chapter provide additional detail on the system architecture layers.

Disk and File System Layout

It is helpful to have an understanding of a system's disk and file system layout. The following illustration represents the layout of a typical MCS server:



The above illustration shows a set of two drives in bays 1 and 2 in a RAID 1 configuration. These drives house the operating system and MCS software. The drives in bays 3 - 8 are configured in a RAID 5 group for the purpose of storing and streaming the transcoded media in the /cache directory.

The following table presents contents of each volume:

Physical Volumes (pv)	Volume Groups (vg)	Logical Volumes (lv)	Directory	Content
sda1			/boot	RHEL boot partition
sda2			(see notes below)	MCS databases
sda3	icps	swap	/dev/dm-0	swap space
		root	/	RHEL system partition
sdb1	ics	cache	/cache	MCS file cache

Note the following:

- sda1 is a standard Linux partition created by RHEL during installation of the operating system.
- sda2 is a 20GB partition that is created on all MCS servers. In a single server configuration, this partition remains empty. The purpose of creating this directory on a single server is to allow for expansion into a cluster configuration without needing to re-image the server. In a cluster configuration, the sda2 partition hosts the PostgreSQL (UMS, ACS, ICS) and MongoDB (messaging) databases. Clusters use DRBD to replicate the /dev/drbd1 directory between the master and slave nodes.
- sda3 contains the system swap disk and the root partition.
- sdb1 is the RAID 5 cache volume used to store transcoded media and various other temporary files.

Workflows that Require RAID 5

The following configurations require a RAID 5 volume as a temporary file cache:

- Workflows that include the file-based method of playback.

Media is not created on the cache while in frame-based playback mode.

- Installations that intend to stream media to iOS or Android mobile devices. In this case, the media on Avid shared storage is transcoded to MPEG-TS (MPEG-2 Transport Stream) and stored locally in the MCS server's `/cache` directory.
- Any installation that includes a multicam workflow. The cache is populated with JPEG images when multi-cam logging is performed. This includes Media Composer Cloud installations that use multicam.
- Interplay | MAM deployments require a RAID 5 cache volume when registered browse proxies include formats that cannot be natively loaded by the Adobe Flash-based player. That is, for non MP4 (H.264/AAC) browse proxies (such MPEG-1, Sony XDCAM, MXF, and WMV), media on proxy storage is transcoded to FLV and stored.
- The Remote Playback workflow introduced with MCS v2.5 for Interplay |MAM caches JPEGs and audio media for frame-based playback.

The following configurations require a cache volume, but do not require RAID 5:

- Media Composer Cloud installations cache media locally on the client systems and do not generally require a RAID 5. The exception to this rule relates to Cloud configurations that use multicam media. The multicam media is converted to a single stream on the MCS server prior to delivery to the client.
- Media Distribute installations.

In Interplay Central v1.5 a RAID 5 cache was required for multi-cam, iOS, and MAM non-h264 systems only. As of Interplay Central v1.6, a separate cache is required for all deployment types, but it does not always need to be RAID 5.

Networking

Regardless of the configuration (single-server or cluster), most MCS servers communicate through a single enabled network interface. This is true even if the server has multiple network ports or adapters.

The exception to this rule is configuring MCS for Interplay MAM where Avid supports "port bonding". Port bonding (also known as link aggregation) combines multiple physical interfaces into a single logical interface. In Interplay MAM deployments, port bonding improves playback performance when multiple clients are making requests of the MCS server simultaneously. With port bonding, more concurrent playback requests can be sustained by a single server, especially for file-based playback. For more information on port bonding, see "Port Bonding for Interplay MAM" in the *Avid MediaCentral Platform Services Installation and Configuration Guide*.

Single-server systems communicate through unicast messaging where a host sends a network packet to another specific host; whereas an MCS cluster uses a combination of unicast and multicast messaging. With unicast messaging, if a single host needs to send the same packet to more than one host, multiple individual messages must be sent. With multicast messaging, a single packet can be sent to a group of hosts simultaneously. This can have advantages in some situations, but it lacks the precision of a point-to-point unicast message.

Required IP Addresses for an MCS Cluster:

While single-server systems generally only require a single IP address, clustered systems require multiple addresses:

- Node IP Address (unicast)

  Every node in an MCS system is assigned a static IP. This is true of both single-server and cluster configurations. While single-server MCS systems support assigning the node IP address through DHCP, clusters require static IP addresses for each node. Network level firewalls and switches must allow the nodes to communicate with one another.

- Virtual IP Address (unicast)

  During the configuration process, a unicast IP address is assigned to the cluster. This IP is associated with a virtual hostname in the site's DNS system. Clients use these virtual identifiers to communicate with the cluster. If a cluster node is offline, clients are still able to communicate with the cluster using the virtual host name (FQDN).

  The virtual IP address is managed by the cluster in the form of the AvidClusterIP resource. It is owned by the master node and moves to the slave node in the event of a failover.

- Cluster IP Address (multicast by default)

  During the configuration process, a multicast IP address is also assigned to the cluster. The multicast address is used for inter-cluster communication. If cluster nodes are spread across multiple network switches, the switches must be configured to allow this multicast traffic. During the cluster configuration, a default multicast IP of 239.192.1.1 can be used as long as no other multicast traffic exists on the network. Alternatively, your IT department can assign a specific multicast address to avoid cross-communication between multicast groups. If your site is not configured to use multicast, a static IP address can be used. However, this requires additional configuration.

Reviewing the IP Addresses:

Once the server is configured, you can use the `ifconfig` command to review the network configuration. The following is an example from a master node of a cluster on HP hardware:

```
[root@wavd-mcs01 ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:60:DD:45:15:21
          inet addr:192.168.10.51  Bcast:192.168.10.255  Mask:255.255.255.0
          inet6 addr: fe40::222:dddd:ff13:1210/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:586964290 errors:0 dropped:0 overruns:0 frame:0
          TX packets:627585183 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:101260694799 (94.3 GiB)  TX bytes:174678891394 (162.6 GiB)
          Interrupt:103

eth0:cl0  Link encap:Ethernet  HWaddr 00:60:DD:45:15:21
          inet addr:192.168.10.50  Bcast:192.168.10.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          Interrupt:103

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:139012986 errors:0 dropped:0 overruns:0 frame:0
          TX packets:139012986 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:101973025015 (94.9 GiB)  TX bytes:101973025015 (94.9 GiB)
```

HP servers identify network adapters with an "eth" prefix whereas Dell servers identify the adapters with an "em1", "p1p1"or "p2p1".

The following is true for the example above:

- "eth0" represents the system's primary physical network interface. Each server will have at least one (usually only one) interface associated with a unicast IP address. In this example, "192.168.10.51" is the IP address for this node. This physical adapter has a state of "UP" which means the adapter is available and active.
- "eth:cl0" (or "cluster 0") is the virtual IP address of the cluster and only exists in cluster configurations. This interface also only appears on the master node that owns the AvidClusterIP resource. In this example "192.168.10.50" is the virtual unicast IP address for the cluster. This virtual adapter has a state of "UP".
- "lo" is the loopback adapter. Each server will have a listing for this. If external network cable(s) are disconnected, the loopback adapter is used by the system to communicate with itself. Without this virtual adapter, some basic system functions would be unable to communicate internally. This virtual adapter has a state of "UP".

The multicast address used for inter-cluster communication does not appear within ifconfig. That address can be verified in the cluster configuration file (corosync.conf) located at: `/etc/corosync/`.

RabbitMQ

RabbitMQ is the message broker ("task queue") used by the MCS top level services. In a single-server configuration, all queues are managed by the server and there is no queue duplication. In a cluster configuration, MediaCentral makes use of RabbitMQ in an active/active configuration with all queues mirrored on exactly two nodes. If there are more than two nodes in the cluster, the RabbitMQ queue is first created on the node that the client or service makes an AMQP connection to. The mirror queue can be located on any other server in the RabbitMQ cluster. Server selection is based on internal RabbitMQ load balancing logic.

The RabbitMQ cluster operates independently of and differently than the Corosync cluster. Nodes in a Corosync cluster assume a master, slave, or load-balancing role, while RabbbitMQ cluster nodes are all active participants. These are known as "disc nodes" in RabbitMQ parlance. This configuration increases system reliability during failover events.

In a multi-zone environment, RabbitMQ nodes in one zone are connected or "federated" with the nodes in the other zones which enables message handling between the zones.

Note the following regarding RabbitMQ clusters:

- All RabbitMQ servers in the cluster are active and can accept connections.
- Any client can connect to any RabbitMQ server in the cluster and access all data.
- In the event of a failover, clients should automatically reconnect to another node.
- The MCS installation scripts create the RabbbitMQ cluster without the need for human intervention.
- MediaCentral v2.4 and later sets partition handling to "autoheal" while earlier versions of the software set partition handling to "ignore".

Queues

A RabbitMQ "queue" is essentially a container created within the RabbitMQ message broker responsible for storing and processing "messages" between connected systems such as MediaCentral UX clients and services on the MCS servers. Messages can be as simple as a string of text that is passed between two systems or it could represent a more complex set of tasks that are executed by multiple systems.

Through clustering, queues are created on a host node and are replicated or "mirrored" to a second node. In the event that the original host node is lost, the queue and related messages are maintained and processed on the mirrored node.

The RabbitMQ Cookie

A notable aspect of the RabbitMQ cluster is the special *cookie* it requires, which allows RabbitMQ on the different nodes to communicate with each other. The RabbitMQ cookie must be identical on each machine, and is set, by default, to a predetermined hard-coded string.

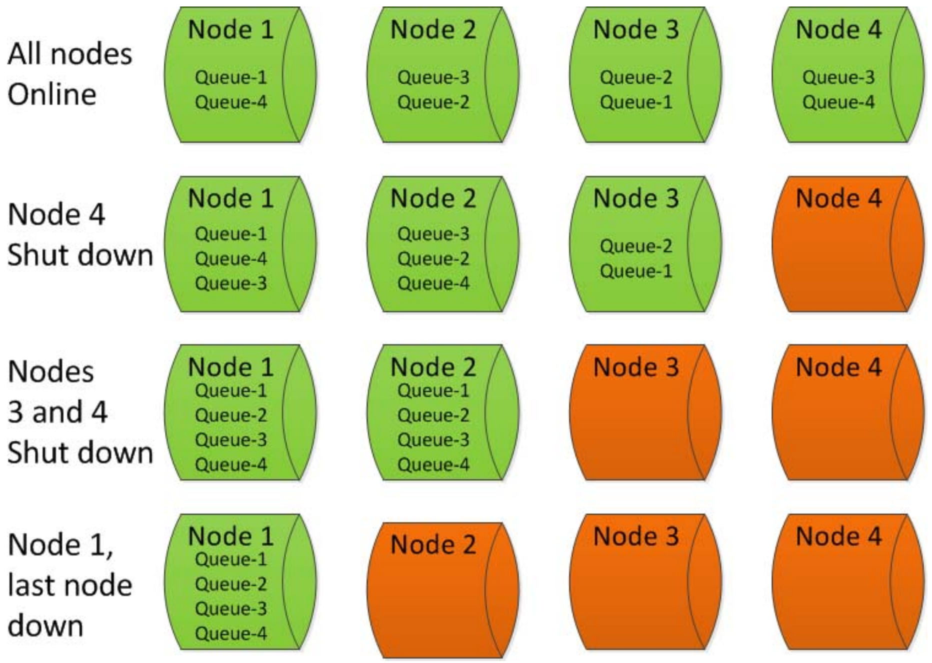
Handling Network Disruptions

- RabbitMQ does not handle network disruptions well. If the network is disrupted on only some of the machines and then it is restored, you should shutdown the machines that lost the network and then power them back on. This ensures they re-join the cluster correctly. This happens rarely, and mainly if the cluster is split between two different switches and only one of them fails.

- On the other hand, if the network is disrupted to *all* nodes in the cluster simultaneously (as in a single-switch setup), no special handling should be required.

Powering Down and Rebooting

When powering down, rebooting, or stopping the RabbitMQ services, the **last node down** must always be the **first node up**. For example, if "Node1" is the last node you stop, it must be the first node you start. Since all queues are replicated to exactly two nodes, queue messages could be lost if the nodes are not shut down properly. Consider the following example (queue numbers are arbitrary):



Note that each queue is replicated on exactly two nodes. When only one nodes remains, only one copy of each queue is present as there are no additional nodes to host the queue replication.

Since Node 1 was the last node down in this example, it is the owner of all RabbitMQ queues and must be the first node to come back up. If Node 2 is powered-on before Node 1, it will look for Node 1 during the startup phase. If it cannot find Node 1, RabbitMQ will not start properly.

Nodes will wait 30 seconds for the "last down" node to become available. If the node is not available in this time-frame, the RabbitMQ cluster will not start.

For more information on shutting down a single node or an entire cluster, see the System Administration chapter starting on page 87.

Suggestions for Further Reading

- Introduction to RabbitMQ: https://www.rabbitmq.com/tutorials/tutorial-one-python.html
- Clustering: http://www.rabbitmq.com/clustering.html
- Mirrored queues: http://www.rabbitmq.com/ha.html
- Network Partitions: http://www.rabbitmq.com/partitions.html

MongoDB

MongoDB is a NoSQL database used by MCS to pass messaging data between services and systems. Mongo can quickly process large amounts of data in a scalable format and has been a core component of Avid MediaCentral Platform Services since version 1.5.0.

Sharded MongoDB

Avid MediaCentral Platform Services v2.6 introduced a second MongoDB database that runs in a "sharded" configuration. This database runs in parallel to the preexisting MongoDB database found in previous versions of the MCS software. The new sharded database is used by the avid-iam service to enable authentication of both Avid and third party "plugins" or "applications" in MediaCentral through a collection of APIs called the Avid Connectivity Toolkit.

The interface between the avid-iam service and the sharded Mongo database is made possible through three new Mongo services: mongod-iam-config-28001, mongod-iam-shard*x*-2710*x*, mongos-iam-27018. The numbers next to each service name represent the network ports used by each service. Multi-zone configurations maintain additional "mongod-iam-shard" services to accommodate the shards from each remote zone.

MCS v2.9 introduces a third MongoDB database which also runs in a sharded configuration for use with the avid-asset and avid-asset-gc services. These services provide a back-end for the "Mixed Sequence Editing" Technology Preview available in v2.9 and provide an infrastructure for future feature development.

The interface between the avid-asset services and the sharded Mongo database is made possible through three new Mongo services: mongod-asset-config*x*-28201, mongod-asset-shard*x*-27200, and mongos-asset-27218. The numbers next to each service name represent the network ports used by each service.

The "x" in the mongod services listed above stands for a numeric value. In multi-zone environments, systems are configured with multiple services with increasing numeric values starting with zero. In the case of "mongod-iam-shardx-2710x", the port number increments with the service.

When first hearing about a "sharded" database, you might think that the database is split into multiple pieces or shards. This is incorrect. In the Avid implementation of sharded MongoDB, each "shard" is a full copy of the database. In a single-server environment, the server maintains a single shard. In a cluster configuration, the master and slave nodes each host a shard. In a multi-zone environment, the servers host shards of the local database as well as shards of the databases from any remote zones. Data is replicated between the servers or zones. These groups of replicated databases are known as "replica sets".

Sharded Mongo provides the following advantages:

- Ability to quickly read and write to local copies of remote databases, omitting network latency.
- Resistance to network disconnection - retaining the ability to read and even to write this zone's attached data.
- High availability of Mongo instances in clustered setup using its own internal mechanism.
- Distribution of read load between 2 nodes in the cluster, increasing overall platform performance.
- Possibility to scale system horizontally (requires additional nodes).

Configuration of sharded Mongo is required for all new installations and upgrades to MCS v2.6.0 and later.

Configuration Options

The sharded MongoDB configuration varies based on your environment. This section describes some of the common MCS configurations and how sharded MongoDB is deployed in those situations.

For more information on configuring sharded MongoDB, see the "Sharded MongoDB" chapter of the *MediaCentral Platform Services Installation and Configuration Guide*.

Single Server Deployments

In this configuration, MongoDB exists as a standalone shard. All database activity is maintained on this server. However, the Avid deployment has been engineered to ensure that any future expansion to a multi-zone configuration requires minimal alterations.

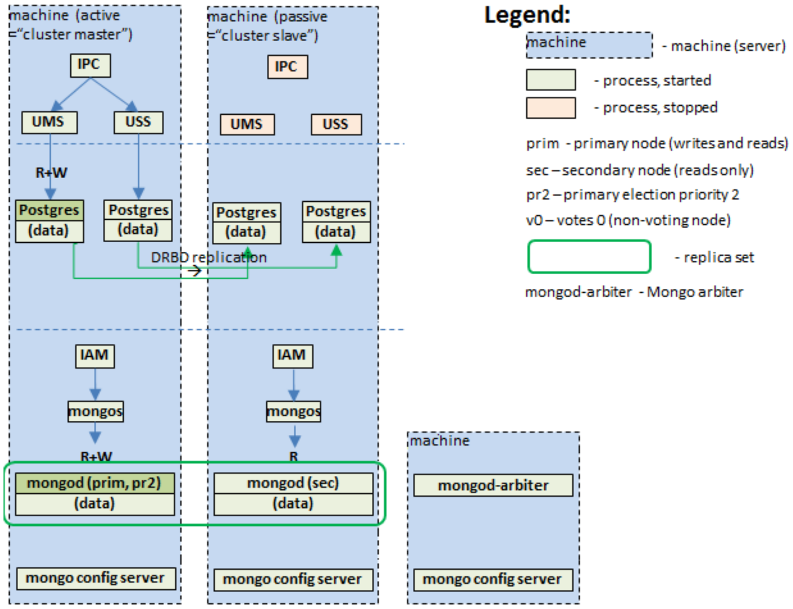
Cluster (2-Node) Deployment (v2.6.0 - v2.8.x)

In local cluster configuration, the master and slave nodes (known as MongoDB "primary" and "secondary" nodes) operate as a replica set where all data is mirrored between the nodes. Message requests are load-balanced between them to increase efficiency. Although copies of the database are stored on the Corosync master and slave nodes, MongoDB uses its own internal mechanisms to provide high availability, separate from Corosync. During a failover, the secondary node becomes the primary.

However, this configuration is slightly more complicated in that MongoDB requires a third system to act as an arbiter. In the event that a node is lost, an election must take place. Since the single remaining server cannot act as a majority, the arbiter acts as a tie-breaking vote.

Arbiters consume a very limited amount of resources on the host system (less than 1% CPU usage). Therefore the CPU, RAM and storage requirements are low and the arbiter can often be co-located on a Linux or Windows system whose primary resources are dedicated to other functions.

The following is an example of a two node cluster configuration with an arbiter:



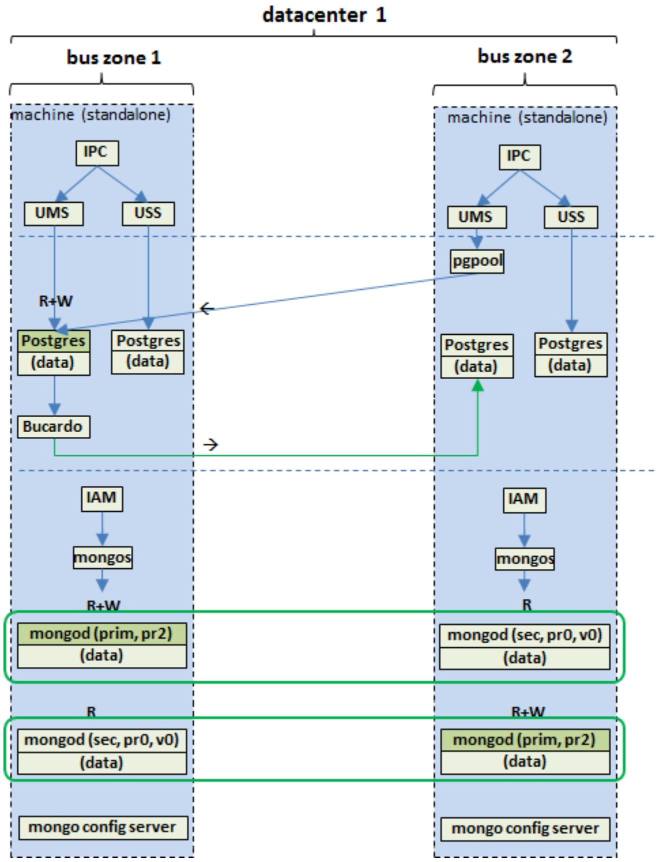
Notice that the arbiter participates in the election process only. It does not maintain a replica of the database.

Cluster (3+ Nodes) Deployment (v2.6.0 - v2.8.x)

This configuration is very similar to the 2-node deployment. In this case, a load-balancing node hosts the arbiter. Even though the arbiter runs on an MCS server in this configuration, the Mongo database is still only maintained on the master and slave nodes.

Multi-Zone - Single Servers (v2.6.0 - v2.8.x)

The following configuration consists of two single-server configurations in a multi-zone environment:



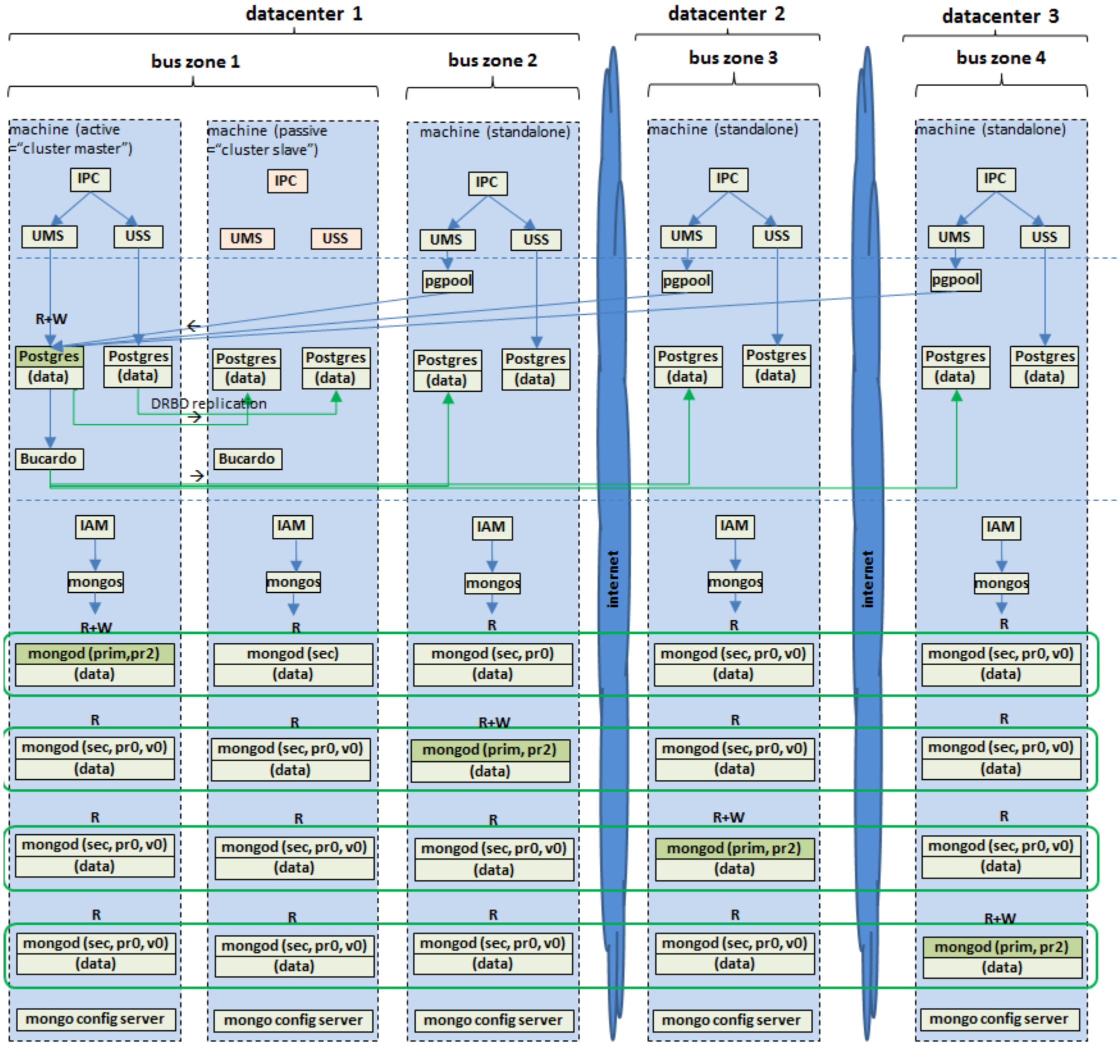
In this configuration, each server has a primary copy of the database. The local database information is replicated on the second node to increase messaging response speeds. If a node is disconnected due to network or power loss, the node can still write to the local copy of the database.

In this example, both zones are located within the same facility (data center). However, the configuration would function the same if the two zones where in geographically different locations. The only additional complexity is to ensure that the correct network ports are open between the facilities.

For more information on MCS network ports, see the Avid Networking Port Usage Guide on the Avid Knowledge Base.

Multi-Zone - Mixed Environment (v2.6.0 - v2.8.x)

The following configuration consists of three geographically separated data centers in a multi-zone configuration consisting of four zones. Data center 1 includes a 2-node cluster (zone 1) as well as a single-server (zone 2). Data center 2 (zone 3) and 3 (zone 4) are also single servers.



Although this is a more complex configuration, it is similar to the "Multi-Zone - Single Server" environment. Each zone maintains a primary shard (and a secondary shard in cluster configurations) of the database as well as replicated shards of every other zone's database. The only difference in this configuration is the number of times that the databases are replicated.

Remember, a "shard" is just another way of saying "copy" in a sharded MongoDB environment.

Sharded MongoDB for v2.9

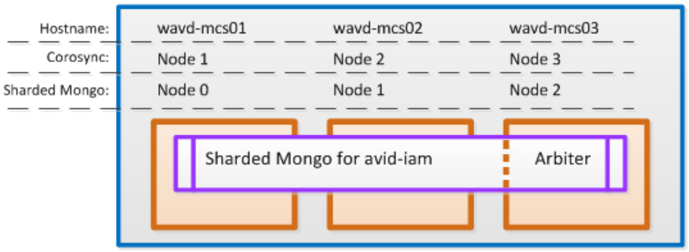
MCS v2.9 introduced two new services (avid-asset and avid-asset-gc) which provide an infrastructure for current and future features. These services require a separate instance of the sharded MongoDB database which is configured somewhat differently than the original sharded MongoDB deployment for avid-iam.

The primary difference between the original avid-iam sharded Mongo deployment and the avid-asset deployment applies to multi-zone configurations. The sharded Mongo configuration for avid-asset requires the creation of an arbiter, even in a multi-zone environment.

To help understand the differences in the configuration, review the following illustrations which outline multiple deployment types. Tables indicating the services running on the nodes have been provided for additional detail. In multi-zone environments, take careful note of the numbers included in each service such as "mongod-asset-shard0" and "mongod-asset-config0".

Configuration #1 - Single Zone v2.8

This configuration depicts a single zone deployment with MCS v2.8 consisting of three nodes. Since this is a local cluster, an arbiter is required to provide a tiebreaker vote in the event of an election.

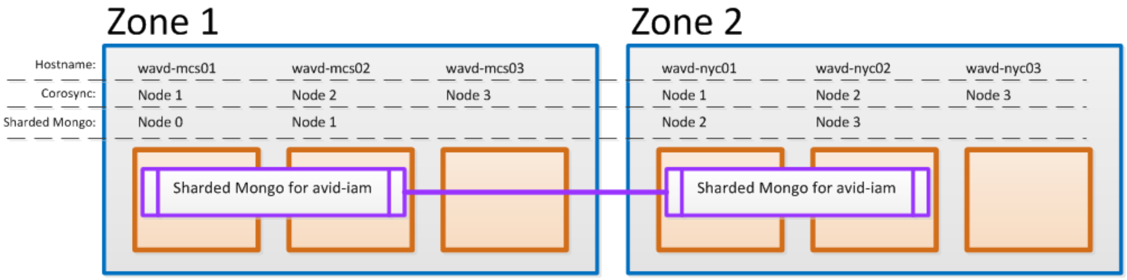


The following services are present in this configuration:

	Node0 - Primary	Node1 - Secondary	Node2 - Arbiter
Sharded Mongo for avid-iam	mongod-iam-shard0-27100 mongod-iam-config-28001 mongos-iam-27018	mongod-iam-shard0-27100 mongod-iam-config-28001 mongos-iam-27018	mongod-iam-shard0-27100 mongod-iam-config-28001

Configuration #2 - Multi-Zone v2.8

This configuration depicts a multi-zone environment consisting of two 3-node clusters. Because the avid-iam deployment of sharded Mongo does not require an arbiter, "Node 3" of each cluster is not included in the sharded Mongo configuration.



The following services are present in this configuration:

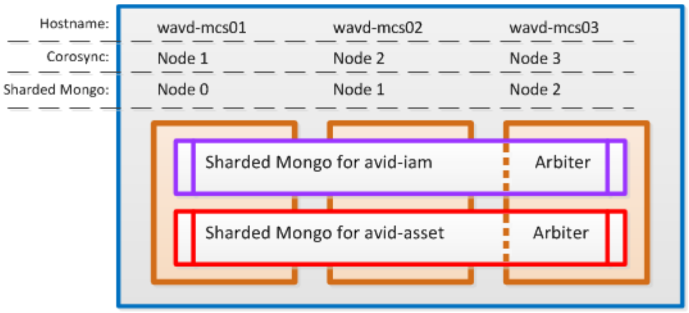
ZONE 1	Node0 - Primary	Node1 - Secondary	(wavd-mcs03)
Sharded Mongo for avid-iam	mongod-iam-shard0-27100 mongod-iam-shard1-27101 mongod-iam-config-28001 mongos-iam-27018	mongod-iam-shard0-27100 mongod-iam-shard1-27101 mongod-iam-config-28001 mongos-iam-27018	

ZONE 2	Node2 - Primary	Node3 - Secondary	(wavd-nyc03)
Sharded Mongo for avid-iam	mongod-iam-shard0-27100 mongod-iam-shard1-27101 mongod-iam-config-28001 mongos-iam-27018	mongod-iam-shard0-27100 mongod-iam-shard1-27101 mongod-iam-config-28001 mongos-iam-27018	

Note that each server hosts a local and remote replica shard from each zone through the “mongod-iam-shard” service.

Configuration #3 - Single Zone v2.9

The third illustration is the same as Configuration #1, but the cluster has been upgraded to MCS v2.9. Notice that the servers now include two separate sharded MongoDB databases.



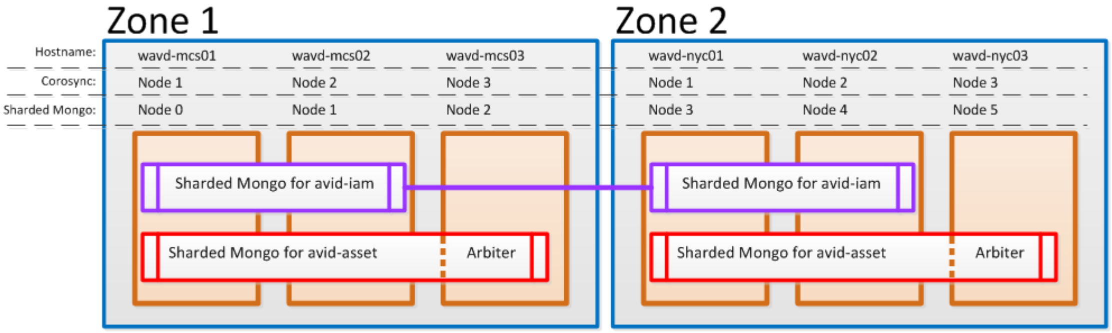
Hostname:	wavd-mcs01	wavd-mcs02	wavd-mcs03
Corosync:	Node 1	Node 2	Node 3
Sharded Mongo:	Node 0	Node 1	Node 2

The following services are present in this configuration:

	Node0 - Primary	Node1 - Secondary	Node2 - Arbiter
Sharded Mongo for avid-iam	mongod-iam-shard0-27100 mongod-iam-config-28001 mongos-iam-27018	mongod-iam-shard0-27100 mongod-iam-config-28001 mongos-iam-27018	mongod-iam-shard0-27100 mongod-iam-config-28001
Sharded Mongo for avid-asset	mongod-asset-shard0-27200 mongod-asset-config0-28201 mongos-asset-27218	mongod-asset-shard0-27200 mongod-asset-config0-28201 mongos-asset-27218	mongod-asset-shard0-27200 mongod-asset-config0-28201

Configuration #4 - Multi-Zone v2.9

The fourth illustration depicts a multi-zone environment for MCS v2.9. The original sharded Mongo configuration for avid-iam maintains shards on the master (Mongo primary) and slave (Mongo secondary) nodes of each zone. Since avid-iam is multi-zone aware, no arbiter is required for that service. However the new instance of sharded Mongo for the avid-asset service is not fully multi-zone compliant and a 2-node + arbiter configuration is required in each zone.



Notice that the sharded Mongo node numbers in this image are sequential (0 through 5). Sequential numbering is normal for new installations, but upgrades might introduce addition nodes to the sharded Mongo ansible hosts file out of order. For instance if a fourth node was added to Zone 1 and the `mongo-create-configuration` script was run, the node would appear in the ansible hosts file as Node6, but the node would be listed under Node 2 as in the following example:

```
[shards]
shard0 shard_tag=region-0
shard1 shard_tag=region-1

[mcs_servers]
node0 ansible_host=wavd-mcs01
node1 ansible_host=wavd-mcs02
node2 ansible_host=wavd-mcs03
node6 ansible_host=wavd-mcs04
node3 ansible_host=wavd-nyc01
node4 ansible_host=wavd-nyc02
node5 ansible_host=wavd-nyc03
```

When running the `mongo-create-configuration` script in an existing configuration, the most important thing to verify is that none of the original node#'s are altered.

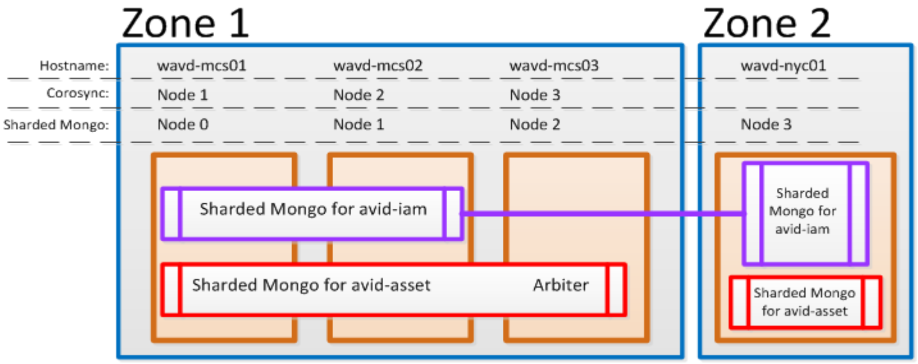
The following services are present in this configuration:

ZONE 1	Node0 - Primary	Node1 - Secondary	Node2 - Arbiter
Sharded Mongo for avid-iam	mongod-iam-shard0-27100 mongod-iam-shard1-27101 mongod-iam-config-28001 mongos-iam-27018	mongod-iam-shard0-27100 mongod-iam-shard1-27101 mongod-iam-config-28001 mongos-iam-27018	
Sharded Mongo for avid-asset	mongod-asset-shard0-27200 mongod-asset-config0-28201 mongos-asset-27218	mongod-asset-shard0-27200 mongod-asset-config0-28201 mongos-asset-27218	mongod-asset-shard0-27200 mongod-asset-config0-28201

ZONE 2	Node3 - Primary	Node4 - Secondary	Node5 - Arbiter
Sharded Mongo for avid-iam	mongod-iam-shard0-27100 mongod-iam-shard1-27101 mongod-iam-config-28001 mongos-iam-27018	mongod-iam-shard0-27100 mongod-iam-shard1-27101 mongod-iam-config-28001 mongos-iam-27018	
Sharded Mongo for avid-asset	mongod-asset-shard1-27200 mongod-asset-config1-28201 mongos-asset-27218	mongod-asset-shard1-27200 mongod-asset-config1-28201 mongos-asset-27218	mongod-asset-shard1-27200 mongod-asset-config1-28201

Configuration #5 - Multi-Zone v2.9

When possible, the sharded Mongo configuration script creates the arbiter in a local zone. The following configuration consists of a three node cluster in Zone 1 and a single server in Zone 2. Notice that the arbiter for avid-asset is located on Zone 1's load-balancing node.



The following services are present in this configuration:

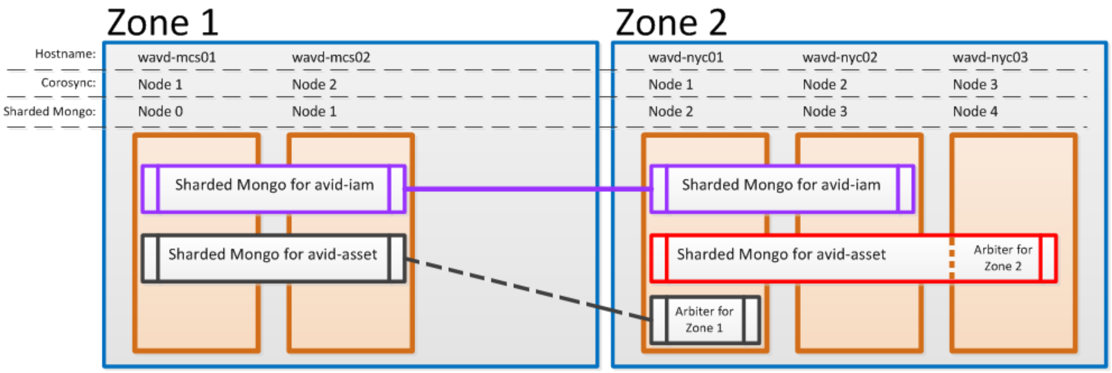
ZONE 1	Node0 - Primary	Node1 - Secondary	Node2 - Arbiter
Sharded Mongo for avid-iam	mongod-iam-shard0-27100 mongod-iam-shard1-27101 mongod-iam-config-28001 mongos-iam-27018	mongod-iam-shard0-27100 mongod-iam-shard1-27101 mongod-iam-config-28001 mongos-iam-27018	
Sharded Mongo for avid-asset	mongod-asset-shard0-27200 mongod-asset-config0-28201 mongos-asset-27218	mongod-asset-shard0-27200 mongod-asset-config0-28201 mongos-asset-27218	mongod-asset-shard0-27200 mongod-asset-config0-28201

ZONE 2	Node3 - Primary
Sharded Mongo for avid-iam	mongod-iam-shard0-27100 mongod-iam-shard1-27101 mongod-iam-config-28001 mongos-iam-27018
Sharded Mongo for avid-asset	mongod-asset-shard1-27200 mongod-asset-config1-28201 mongos-asset-27218

Configuration #6 - Mult-Zone v2.9

In this configuration, Zone 1 consists of only two nodes and Zone 2 is a three node cluster. Since additional MediaCentral servers are not available in Zone 1, the sharded Mongo configuration script is programed to install the arbiter on the primary node of the remote zone. This avoids the need to create a separate local arbiter in Zone 1. The arbiter for Zone 1 is represented as the “mongod-asset-shard0-29200” and “mongod-asset-config0-30201”services on the primary node in Zone 2.

When the arbiter is located on a remote zone, the ports associated with the “mongod-asset-shard0” and “mongod-asset-config0” services are changed to 29200 and 30201. The port numbers increment if required to accommodate additional zones.



The following services are present in this configuration:

ZONE 1	Node0 - Primary	Node1 - Secondary
Sharded Mongo for avid-iam	mongod-iam-shard0-27100 mongod-iam-shard1-27101 mongod-iam-config-28001 mongos-iam-27018	mongod-iam-shard0-27100 mongod-iam-shard1-27101 mongod-iam-config-28001 mongos-iam-27018
Sharded Mongo for avid-asset	mongod-asset-shard0-27200 mongod-asset-config0-28201 mongos-asset-27218	mongod-asset-shard0-27200 mongod-asset-config0-28201 mongos-asset-27218

ZONE 2	Node2 - Primary	Node3 - Secondary	Node4 - Arbiter
Sharded Mongo for avid-iam	mongod-iam-shard0-27100 mongod-iam-shard1-27101 mongod-iam-config-28001 mongos-iam-27018	mongod-iam-shard0-27100 mongod-iam-shard1-27101 mongod-iam-config-28001 mongos-iam-27018	
Sharded Mongo for avid-asset	mongod-asset-shard0-29200 mongod-asset-shard1-27200 mongod-asset-config0-30201 mongod-asset-config1-28201 mongos-asset-27218	mongod-asset-shard1-27200 mongod-asset-config1-28201 mongos-asset-27218	mongod-asset-shard1-27200 mongod-asset-config1-28201

In summary, MediaCentral Platform Services v2.9 includes three different MongoDB databases:

- Original non-sharded MongoDB introduced in MCS v1.5
- Sharded MongoDB for avid-iam introduced in MCS v2.6
- Sharded MongoDB for avid-asset introduced in MCS v2.9

Suggestions for Further Reading

- MongoDB Sharding Introduction: https://docs.mongodb.org/manual/sharding/
- MongoDB Sharding FAQ: https://docs.mongodb.org/manual/faq/sharding/

Playback Services

As previously described, MediaCentral Platform Services (MCS) is a collection of services running on one or more servers, providing a base infrastructure for multiple solutions. One of the primary services on an MCS system is the playback service which provides playback for a range of media assets to a variety of platforms.

Frame vs File-Based Playback

Earlier versions of MediaCentral UX used only frame-based playback. Starting with v2.1, MediaCentral UX includes an option to use file-based playback.

Both MediaCentral UX and Interplay MAM allow the end-user to configure the session for either playback method. Frame-based playback is more CPU-intensive in MAM configurations that use an MP4 format with an H.264/AAC essence. In file-based playback in Interplay Production configurations, assets are converted to FLV/MP3 files on the MCS server. The first playback transcodes the file and initially consumes more resources than frame-based playback.

For a detailed description of frame and file-based playback, see “Selecting Frame-Based Playback or File-Based Playback” in the *Avid MediaCentral | UX User’s Guide*.

For information on configuring file-based playback, see “Configuring File-Based Playback” in the *Avid MediaCentral | UX Administration Guide*.

Playback for Mobile Devices

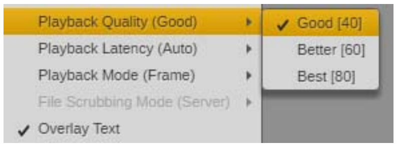
In multi-resolution environments, mobile workflows create reusable files on the server that point to the high resolution media to ensure the best image quality for the WiFi-stream. Information on low-res proxy formats is only referenced in the event that high-res media is unavailable.

Playback for Web Browsers and the MediaCentral UX Desktop App

In multi-resolution environments, media playback for the browser points to the low resolution media to maximize stream and client counts. Information on high-res formats is only referenced in the event that low-res proxy media is unavailable.

The default playback method for MediaCentral UX is frame-based. In this mode, media is encoded as a series of JPEG files that are streamed directly from the server for playback. This technique provides frame-accuracy and a smooth transition between cuts. Frame-based playback is high quality, but requires a higher network bandwidth than file-based playback.

Frame-based playback use one of three Playback Quality settings that are selectable by the user through the Media pane’s context menu.



Although these are fixed quality settings, the size of each frame can vary greatly. The two extreme examples are black frames which result in very small files and color noise which result in very large files.

The quality of the video in the Media pane can be adjusted further through the Image Quality Settings in the MediaCentral UX System Settings. For more information, see “Configuring Image Quality” in the *Avid MediaCentral | UX Administration Guide*.

Playback for Interplay MAM

The default playback method for Interplay | MAM is file-based. In this mode, media is encoded as a series of FLV files that are downloaded to a temporary cache on the client workstation for playback. File-based playback provides good quality in low-bandwidth situations.

Asset are often associated with a single FLV file, but that is not always the case. Longer assets might be segmented into multiple files. If an asset is associated with multiple audio tracks (beyond the original stereo pair), the additional tracks are downloaded as separate FLV files – one per stereo pair. These tracks do not include video and are generally formated as MP3 so the file size is negligible.

Sequences (EDLs) that are composed of multiple assets are broken down into multiple FLV files (generally - one per asset). If for example, an EDL consisted of three segments, the client downloads three FLV files as the user plays through it. However, not all FLV files s are downloaded immediately. Only the segment that is currently active and the following segment are requested from the server. Downloading one segment ahead of the current ensures that playback can continue without interruption.

Cluster Specific Systems

The information in this section specifically relates to cluster configurations and does not apply to single-server installations.

Corosync and Pacemaker

Corosync and Pacemaker are independent systems which operate closely together to create the core cluster monitoring and failover capabilities.

Corosync is the messaging layer used by the cluster. Its primary purpose is to maintain awareness of node membership - nodes joining or leaving the cluster. It also provides a quorum system to assist in deciding who takes ownership of a resource if a node is lost.

Pacemaker is a resource manager. A resource represents a service or a group of services that can be manged by the cluster. Pacemaker maintains a configuration file (cib.xml) which defines all resources within the cluster and governs how the resources react to a failure. Examples of these governing rules are: fail-counts, actions to take upon a failure, timeout values and so forth.

During a standard boot process, Corosync starts before Pacemaker to help identify which nodes are available. Pacemaker then identifies which resources need to be started based on the information provided by Corosync. Example: If “node-1” is the first node to be started and it is one of the drbd nodes which hosts the database, the node becomes the master node and Pacemaker starts the appropriate resources.

If a resource fails, Pacemaker will attempt to restart the resource based on the rules configured for that resource within the configuration file. If the resource fails enough times to reach the fail-count threshold, it will no longer attempt to restart it. When a failed resource is operating on the master node of the cluster, a failover to the slave node might occur (depending on the resource).

For more information, see “Interacting with Resources” on page 59 and “Cluster Resource Monitor” on page 80.

Clustering Infrastructure Services

The MCS services and databases depend on a functioning clustered infrastructure. The infrastructure is supported by a small number of open-source software components designed specifically (or very well suited) for clustering. For example, Pacemaker and Corosync work in tandem to restart failed services, maintain a fail-count, and move resources from the master node to the slave node, when failover criteria are met.

The following table presents the services pertaining to the infrastructure of the cluster:

Software	Function	Node 1 (Master)	Node 2 (Slave)	Node 3	Node *n*
RabbitMQ	Cluster Message Broker/Queue	ON	ON	ON	ON
DRBD	Database Volume Mirroring	ON	ON	OFF	OFF
Pacemaker	Cluster Management & Service Failover	ON	ON	ON	ON
Corosync	Cluster Engine Data Bus	ON	ON	ON	ON
GlusterFS	File Cache Mirroring	ON	ON	ON	ON
	🟩 = ON (RUNNING)	🟧 = OFF (STANDBY)		🟥 = OFF (DOES NOT RUN)	

Note the following:

- RabbitMQ, the message broker/queue used by ACS, maintains its own clustering system. It is not managed by Pacemaker.
- DRBD mirrors the MCS databases across the two servers that are in a master-slave configuration. This provides redundancy in case of a server failure.
- Pacemaker: The cluster resource manager. Resources are collections of services participating in high-availability and failover.
- Corosync: The fundamental clustering infrastructure.

- Corosync and Pacemaker work in tandem to detect server and application failures, and allocate resources for failover scenarios.
- GlusterFS mirrors media cached on a RAID 5 volume to all nodes in the cluster; each with their own RAID 5 volume.

MCS Services, Resources and Cluster Databases

The following table lists some of the primary MCS services, the name of the resource that Pacemaker uses to manage those services, and where each runs in a clustered environment:

Service	**Resource Name**	**Node 1 (Master)**	**Node 2 (Slave)**	**Node 3**	**Node *n***
IPC Core Services ("the middleware") (avid-interplay-central)	AvidIPC	🟩 ON	🟧 OFF	🟥 OFF	🟥 OFF
Avid Upstream Service (avid-upstream)	AvidUpstream	🟩 ON	🟧 OFF	🟥 OFF	🟥 OFF
Avid Iam Service (avid-iam) (MCS v2.6 and later)	AvidIam (AvidIamEverywhere)	🟩 ON	🟩 ON	🟥 OFF	🟥 OFF
Avid Asset Service (avid-asset) (MCS v2.9 and later)	AvidAsset (AvidAssetEverywhere)	🟩 ON	🟩 ON	🟥 OFF	🟥 OFF
Avid Asset GC Service (avid-asset-gc) (MCS v2.9 and later)	AvidAssestGc (AvidAssetGcEverywhere)	🟩 ON	🟩 ON	🟥 OFF	🟥 OFF
User Management Service (avid-ums)	AvidUMS	🟩 ON	🟧 OFF	🟥 OFF	🟥 OFF
UMS session cache service (redis)	Redis	🟩 ON	🟧 OFF	🟥 OFF	🟥 OFF
User Setting Service (avid-uss)	AvidUSS	🟩 ON	🟧 OFF	🟥 OFF	🟥 OFF
Avid Common Services bus ("the bus") (avid-acs-ctrl-core)	AvidACS	🟩 ON	🟧 OFF	🟥 OFF	🟥 OFF
Avid Monitor (avid-monitor)	AvidClusterMon	🟩 ON	🟧 OFF	🟥 OFF	🟥 OFF
Avid Service Manager (avid-acs-service-manager) (MCS v2.6 and later)	AvidServiceManager	🟩 ON	🟧 OFF	🟥 OFF	🟥 OFF
Playback Service (avid-icps-manager)	AvidICPS (AvidICPSEverywhere)	🟩 ON	🟩 ON	🟩 ON	🟩 ON
Avid Gateway (avid-acs-gateway) (MCS v2.5 and later)	AvidGateway (AvidGatewayEverywhere)	🟩 ON	🟩 ON	🟩 ON	🟩 ON
Avid Nginx (nginx) (MCS v2.5 and later)	AvidNginx (AvidNginxEverywhere)	🟩 ON	🟩 ON	🟩 ON	🟩 ON
Load-Balancing Services ("the back-end") (avid-all)	AvidAll (AvidAllEverywhere)	🟩 ON	🟩 ON	🟩 ON	🟩 ON
🟩 = ON (RUNNING)	🟧 = OFF (STANDBY)		🟥 = OFF (DOES NOT RUN)		

Note the following:

- All MCS services run on the Master node in the cluster.
- Some MCS services are run on the Slave node in standby only. These services are started automatically during a failover.
- Starting with MCS v2.6.1, AvidIam runs on the master and slave nodes as a cluster resource.
- Other services spawned by the Avid Common Service bus run on all nodes. The Playback Service (ICPS) is an example of such a service. It runs on all nodes for scalability (load-balancing supports many concurrent clients and/or large media requests) and high availability (service is always available).

The following table lists the bus-dependent services:

Services and Resources	**Node 1 (master)**	**Node 2 (slave)**	**Node 3**	**Node *n***
AAF Generator* (avid-aaf-gen)	ON	ON	ON	ON
MCS Messaging (avid-acs-messenger & avid-acs-mail)	ON	ON	ON	ON

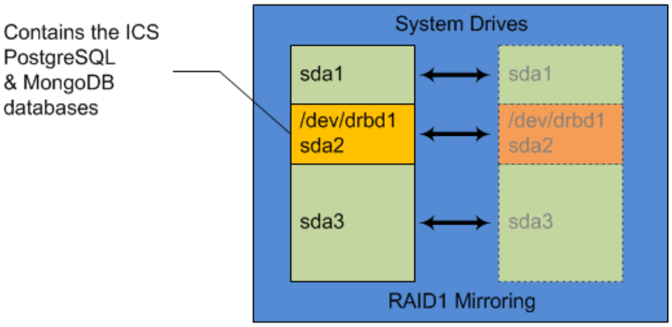
* The AAF Generator runs on all nodes. However, since it is used by the MCS Core Service (“the middleware”), it is only in operation on the master and slave nodes.

The following table lists the MCS databases, and where they run:

MCS Databases		**Node 1 (Master)**	**Node 2 (Slave)**	**Node 3**	**Node *n***
ICS Database	PostgreSQL	ON	OFF	OFF	OFF
Service Bus Messaging Database	MongoDB	ON	OFF	OFF	OFF
Sharded Mongo for avid-iam	Sharded MongoDB	ON	ON	OFF	OFF
Sharded Mongo for avid-asset	Sharded MongoDB	ON	ON	OFF	OFF
RabbitMQ database	Mnesia	ON	ON	ON	ON
🟩 = ON (RUNNING)		🟧 = OFF (STANDBY)		🟥 = OFF (DOES NOT RUN)	

DRBD and Database Replication

The system drive on a typical MCS server consists of three partitions: sda1, sda2 and sda3. As shown in the following diagram, the sda2 partition in a cluster is used for storing the MCS databases such as PostgreSQL and MongoDB.



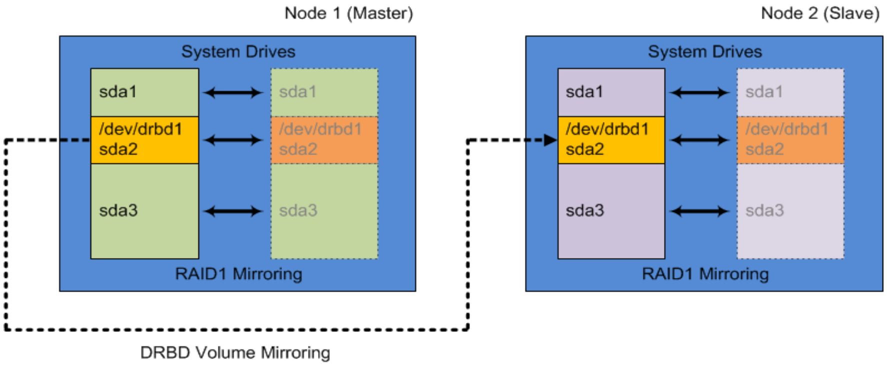
The following table details the contents of the databases stored on the sda2 partition:

Database	Directory	Contents
MongoDB	/mnt/drbd/mongo_data	Data store for messaging, avid-iam, avid-asset
PostgreSQL	/mnt/drbd/postgres_data	UMS - User Management Services ACS - Avid Common Service bus ICPS - Interplay Central Playback Services MPD - Media Distribute

In a single-server configuration, the PostgreSQL and MongoDB databases are created in `/var/lib/pgsql` *and* `var/lib/mongo`.

MCS uses the open source Distributed Replicated Block Device (DRBD) storage system software to replicate the sda2 partition across the Master/Slave cluster node pair. DRBD runs on the master node and slave node only, even in a cluster with more than two nodes. PostgreSQL maintains the databases on sda2. DRBD mirrors them.

The following illustration shows DRBD volume mirroring of the sda2 partition across the master and slave.



Gluster and Cache Replication

Recall that MCS transcodes media from the format in which it is stored on the Avid shared storage (or standard file system storage) into an alternate delivery format, such as FLV, MPEG-2 Transport Stream, or JPEG image files. In a deployment with a single MCS server, the MCS server maintains a cache where it keeps recently-transcoded media. In the event that the same media is requested again, the MCS server can deliver the cached media, without the need to re-transcode it.

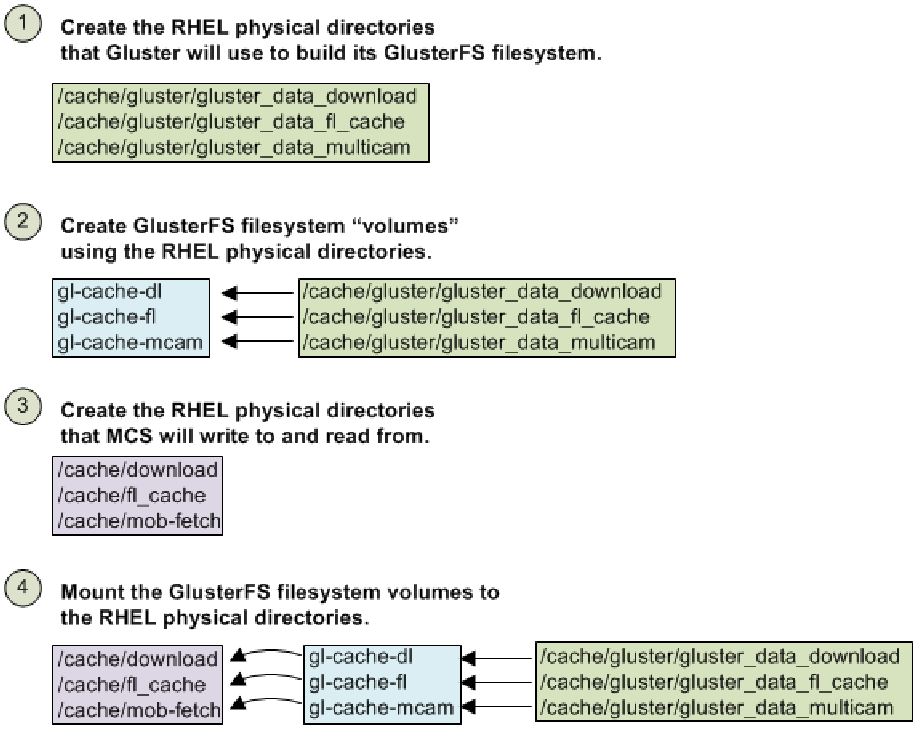
In an MCS cluster, caching is taken one step farther. In a cluster, the contents of the cache volumes are replicated across all the nodes, giving each server access to all the transcoded media. The result is that each MCS server has access to the media transcoded by every other node. When one MCS server transcodes media, the other MCS servers can also make use of it, without re-transcoding.

The replication process is controlled by Gluster, an open source software solution for creating shared file systems. In MCS, Gluster manages data replication using its own highly efficient network protocol. In this respect, it can be helpful to think of Gluster as a “network file system” or even a “network RAID” system.

Gluster operates independently of other clustering services. You do not have to worry about starting or stopping Gluster when interacting with MCS services or cluster management utilities. For example, if you remove a node from the cluster, Gluster itself continues to run and continues to replicate its cache against other nodes in the Gluster group. If you power down the node for maintenance reasons, it will re-synchronize and 'catch up' with cache replication when it is rebooted.

The correct functioning of the cluster cache requires that the clocks on each server in the cluster are set to the same time. See “Configure Date and Time Settings” in the MediaCentral Platform Services Installation and Configuration Guide for details on configuring time sync.

The following illustration summarizes the file system operations as configuring during the installation process:



1 Create the RHEL physical directories that Gluster will use to build its GlusterFS filesystem.

2 Create GlusterFS filesystem “volumes” using the RHEL physical directories.

3 Create the RHEL physical directories that MCS will write to and read from.

4 Mount the GlusterFS filesystem volumes to the RHEL physical directories.

/cache/gluster/gluster_data_download
/cache/gluster/gluster_data_fl_cache
/cache/gluster/gluster_data_multicam
gl-cache-dl
gl-cache-fl
gl-cache-mcam
/cache/gluster/gluster_data_download
/cache/gluster/gluster_data_fl_cache
/cache/gluster/gluster_data_multicam
/cache/download
/cache/fl_cache
/cache/mob-fetch
/cache/download
/cache/fl_cache
/cache/mob-fetch
gl-cache-dl
gl-cache-fl
gl-cache-mcam
/cache/gluster/gluster_data_download
/cache/gluster/gluster_data_fl_cache
/cache/gluster/gluster_data_multicam

Multi-Zone Environments

By default, each MediaCentral system operates independently, within a single "zone", where each zone consists of the following:

- One MediaCentral Platform Services single-server or cluster
- One Interplay Production, iNEWS, and / or Interplay MAM database

A multi-zone environment combines two or more single-zone systems together to enable enhanced WAN workflows. The benefits of a multi-zone environment include:

- Multi-zone user management: Centralized user management across all zones.

  In a multi-zone environment, one zone maintains a master copy of the user database. The master zone has the ability to read and write to database while all slave zones have read-only access. All log-in activity in the slave zones is channeled through the master zone. In the event of a network disruption, the slave zones continues to operate in read-only mode until connectivity to the master zone is re-established.
- Multi-zone central index search: Search across multiple databases in different zones.

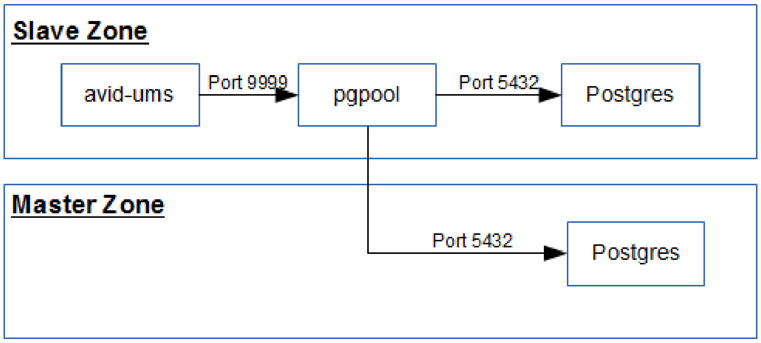
  If Media Index is configured across the multi-zone environment, users can quickly search for assets across all zones and instantly play the material in the remote zone.
- Multi-zone media asset delivery: Transfer the high-resolution material you found on a remote zone through an indexed search to your local zone.

  If users wish to combine remote assets in local sequences, a transfer of the material from the remote zone to the local zone can be initiated.

Systems configured in a multi-zone environment have three additional services

- bucardo (master zone only) - In a multi-zone configuration, bucardo replicates the postgres user database between the master and slave zones. This replication process verifies that the slave zone has the most recent user database in the event that it is disconnected from the master zone.
- pgpool and pgpoolchecker (slave zone only) - These services manage the connection to the user database. In normal operation, pgpool creates a connection to database in the master zone. If the master zone is disconnected, pgpool redirects the connection to a read-only replica of the postgres user database in the slave zone.

The ums (user management) service does not communicate directly with bucardo, it only communicates with postgres. In the slave zone of a multi-zone configuration, the postgres communication is routed through pgpool. The default postgres port is 5432. pgpool is "listening" on port 9999. Therefore, the ums port must be set to port 9999 if you want to use the "standard" multi-zone configuration. The following graphic illustrates the service communication:



3 Services and Resources

Services are highly important to the operation and health of an MCS system. As noted in "System Architecture" on page 23, services are responsible for all aspects of MCS activity, from the ACS bus, to end-user management and transcoding. Additional services supply the clustering infrastructure. In a cluster, some MCS services are managed by Pacemaker, for the purposes of high-availability and failover readiness. Services overseen by Pacemaker are called *resources*.

Services vs Resources

While single-server systems run only services, clusters feature both Linux *services* and Pacemaker cluster *resources* and it is important to understand the difference between the two. In the context of clustering, a *resource* is simply a Linux service or a group of services managed by Pacemaker. Managing services in this way allows Pacemaker to monitor the services and automatically restart them when they fail. Additionally, Pacemaker can shut down resources on one node and start them on another when a fail-count threshold has been reached. This prevents failing services from restarting infinitely.

It can be helpful to regard a cluster *resource* as Linux *service* inside a Pacemaker "wrapper". The wrapper includes the actions defined for it (*start*, *stop*, *restart*, etc.), timeout values, failover conditions and instructions, and so on. In short, Pacemaker manages resources, not services.

For example, "*avid-interplay-central*" is the core MediaCentral service. Since the platform cannot function without it, this service is overseen and managed by Pacemaker as the *AvidIPC* resource.

The status of a Linux service can be verified by entering a command of the following form at the command line:

```
service <service_name> status
```

In contrast, the status of a cluster resource is verified through the Pacemaker Cluster Resource Manager, *crm*, as follows:

```
crm status <resource>
```

Tables of Services and Resources

The tables in this section provide lists of essential services that need to be running on single-node and clustered configurations. It includes four tables:

- **Single Server**: The services that must be running in a single server deployment.
- **Cluster - Master Node Only**: The services that must be running on the master node only. Although some of these services may be available in standby on the slave node, they should not be actively running on any other node.
- **Cluster - All Nodes**: The services that must be running on all nodes.
- **Cluster - Pacemaker Resources**: The services managed by Pacemaker.

These tables are not exhaustive. They are meant to highlight essential services that operate on a MediaCentral Platform server.

Single Server

The following table presents the services that must be running on the server, in an MCS deployment with only one server.

Service	Description
avid-aaf-gen	AAF Generator service, the service responsible for saving sequences. To reduce bottlenecks when the system is under heavy load, five instances of this service run concurrently, by default.
avid-acs-ctrl-core	Avid Common Service bus ("the bus") Includes essential bus services needed for the overall platform to work: • "boot" service (provides registry services to bus services) • "attributes" service (provides system configuration of IPC) • "federation" service (initializes multi-zone configurations) The *avid-acs-ctrl-core* service is a critical service. The following services will not start or function correctly if *avid-acs-ctrl-core* is not running. • avid-icps-manager • avid-ums • avid-interplay-central • avid-all • avid-acs-messenger • avid-mpd
avid-acs-gateway	Added in MCS v2.5, this service was created to provide a common access point for the main bus services (registry, federation, attributes and infrastructure). By directing services to the gateway, connections to the bus access layer (BAL) are standardized.

Service	Description
avid-acs-messenger	The services related to the IPC end-user messaging feature: • “messenger” service (handles delivery of user messages) • “mail” service (handles mail-forwarding feature) This service registers itself on the ACS bus. All instances are available for handling requests, which are received by way of the bus via a round-robin-type distribution system.
avid-all	Encapsulates all ICPS back-end services: • avid-config • avid-isis • avid-fps • avid-jips • avid-spooler • avid-edit
avid-asset	To enable the “Mixed Sequence Editing” Technology Preview added in MCS v2.9, the avid-asset service stores temporary (draft) sequence information in the sharded Mongo database. These “temp” sequences remain in the database until the user completes the editing and conforms the mixed-asset sequence. In addition to storing the temp sequences, the avid-asset service is responsible for marking sequences that have been conformed for deletion.
avid-asset-gc	The avid-asset-gc service works in conjunction with the avid-asset service and is responsible for the actual deletion of the data. This process is not immediate and instead occurs on a scheduled basis
avid-ccc (if installed)	Closed Captioning service (requires separate installation)
avid-iam	The Identity and Access Management service is a foundational platform service, which provides scalable, highly available and performance management of identity (user/context/service/application) data and authorizations This service participates in the MediaCentral UX user authentication (sign in) process. This service has a dependency on the (sharded) mongos-iam services.
avid-interplay-central	IPC Core services (“the middleware”)
avid-mpd (if installed)	Services related to Media Distribute include: • avid-media-central-mpd • avid-mpd • servicemix Operates similarly to the avid-acs-messenger service described above. This service is only available when Media Distribute (separate installer) is installed on the system.
avid-ums	User Management Service
avid-upstream	Required for the API toolkit, the avid-upstream service is a gateway that maps synchronous HTTP request / response calls to asynchronous messaging backed by the ACS Bus.

Service	Description
avid-uss	User Setting Service - enables custom user data such as saved searches, layouts, opened panes and more to be retained between sessions.
mongod	MongoDB database for data from the following services: • ICS Messaging (avid-acs-messenger) data • ACS bus (avid-acs-ctrl-core) registry MCS v2.6 introduces a "sharded" Mongo configuration. A single server will host the following services: • mongos-iam-27018 • mongod-iam-config-28001 • mongod-iam-shard0-27100 These services are used in conjunction with the avid-iam service. The numbers listed at the end of the name represent the port used by the service. If the server is part of a multi-zone configuration, additional shard services will exist: • mongod-iam-shard1-27101 • mongod-iam-shard2-27102 • etc. MCS v2.9 adds a second sharded Mongo database to serve as a back-end for the avid-asset service. The following additional services are present: • mongod-asset-config0-28201 • mongod-asset-shard0-27200 • mongos-asset-2721 If the server is part of a multi-zone configuration, additional mongod-asset-config and mongod-asset-shard services will exist. *The original implementation of MongoDB found in prior versions of MCS is maintained in MCS v2.6 and later. Sharded Mongo runs in parallel and is separate from the original MongoDB configuration.*
postgresql-9.1	PostgreSQL database for user management and attributes data.
nginx	Added in MCS v2.5, this is the web server where all client connections are processed. SSL certificate authentication are processed through nginx. Client calls are proxied to either AvidIPC (avid-interplay-central) or the avid-icps-manager for playback.
rabbitmq-server	Messaging broker/queue for the ACS bus.
redis	Redis is a key-value data store used to store user session data. This allows MCS to cache active session data and not continuously make calls to the postgresql database to retrieve user information.

Service	Description
“Media Index services” (if configured)	Services related to the Media Index service include: • avid-acs-search • avid-acs-autocomplete • avid-acs-media-index-configuration • avid-acs-search-import • avid-acs-media-index-feed • avid-acs-media-index-status-provider • avid-acs-media-index-permission • avid-acs-media-index-thesaurus (added in MCS v2.4) • elasticsearch • elasticsearch-tribe • pam-agent (for Interplay PAM configurations) These services are only running when Media Index has been enabled.
“Multi-Zone services” (if configured)	Services related to multi-zone configurations include: • bucardo (master zone) • pgpool and pgpoolchecker (slave zone) These services are only running if a multi-zone configuration has been enabled.

Cluster - Master Node

The following table presents the services that must be running on a cluster master node.

Service	Description
avid-acs-ctrl-core	Avid Common Service bus (“the bus”) Includes essential bus services needed for the overall platform to work: • “boot” service (provides registry services to bus services) • “attributes” services (provides system configuration of IPC) • “federation” service (initializes multi-zone configurations) The *avid-acs-ctrl-core* service is a critical service. The following services will not start or function correctly if *avid-acs-ctrl-core* is not running. • avid-all • avid-acs-messenger • avid-icps-manager • avid-interplay-central • avid-ums
avid-acs-service-manager	Added in MCS v2.5, this service was created to operation statistics about services to the ACS monitor. In MCS v2.5 this service runs on all nodes. In MCS v2.6 the configuration was altered to only run on the cluster master node.
avid-iam	The Identity and Access Management service is a foundational platform service, which provides scalable, highly available and performance management of identity (user/context/service/application) data and authorizations. This service participates in the MediaCentral UX user authentication (sign in) process and runs on the master and slave nodes only. This service has a dependency on the (sharded) mongos-iam services. *Although listed in the “Cluster - Master Node Only” table, this service runs on the master and slave nodes.*
avid-interplay-central	IPC Core services (“the middleware”)
avid-monitor	This service monitors the nodes in the cluster.If a node goes down (network outage, etc.), this service reports the node status to Pacemaker.
avid-asset	To enable the “Mixed Sequence Editing” Technology Preview added in MCS v2.9, the avid-asset service stores temporary (draft) sequence information in the sharded Mongo database. These “temp” sequences remain in the database until the user completes the editing and conforms the mixed-asset sequence. In addition to storing the temp sequences, the avid-asset service is responsible for marking sequences that have been conformed for deletion. This service runs on the master and slave nodes only.
avid-asset-gc	The avid-asset-gc service works in conjunction with the avid-asset service and is responsible for the actual deletion of the data. This process is not immediate and instead occurs on a scheduled basis This service runs on the master and slave nodes only.
avid-ums	User Management Service

Service	Description
avid-upstream	Required for the API toolkit, the avid-upstream service is a gateway that maps synchronous HTTP request / response calls to asynchronous messaging backed by the ACS Bus.
avid-uss	User Setting Service - enables custom user data such as saved searches, layouts, opened panes and more to be retained between sessions.
drbd	DRBD (Distributed Replicated Block Device) is used to mirror the system disk partition containing the two databases from master to slave, for failover readiness: • PostGreSQL • MongoDB DRBD is fully functional on both master and slave. It is included in this table for convenience.
mongod	MongoDB database for data from the following services: • ICS Messaging (avid-acs-messenger) data • ACS bus (avid-acs-ctrl-core) registry MCS v2.6 introduces a “sharded” Mongo configuration. The cluster master node will host the following services: • mongos-iam-27018 • mongod-iam-config-28001 • mongod-iam-shard0-27100 These services are used in conjunction with the avid-iam service. The numbers listed at the end of the name represent the port used by the service. If the server is part of a multi-zone configuration, additional shard services will exist: • mongod-iam-shard2-27101 • mongod-iam-shard3-27102 • etc. MCS v2.9 adds a second sharded Mongo database to serve as a back-end for the avid-asset service. The following additional services are present: • mongod-asset-config0-28201 • mongod-asset-shard0-27200 • mongos-asset-2721 If the server is part of a multi-zone configuration, additional mongod-asset-config and mongod-asset-shard services will exist. *The original implementation of MongoDB found in prior versions of MCS is maintained in MCS v2.6 and later. Sharded Mongo runs in parallel and is separate from the original MongoDB configuration.*
postgresql-9.1	PostgreSQL database for user management and attributes data.
redis	Redis is a key-value data store used to store user session data. This allows MCS to cache active session data and not continuously make calls to the postgresql database to retrieve user information.
avid-ccc (if installed)	Closed Captioning service (requires separate installation)

Service	Description
“Multi-Zone services” (if configured)	Services related to multi-zone configurations include: • bucardo (master zone) • pgpool and pgpoolchecker (slave zone) These services are only running if a multi-zone configuration has been enabled.

Cluster - All Nodes

The following table presents the services that must be running on all nodes in a cluster.

Service	Description
avid-all	Encapsulates all ICPS back-end services: • avid-config • avid-isis • avid-fps • avid-jips • avid-spooler • avid-edit
avid-aaf-gen	AAF Generator service, the service responsible for saving sequences. To reduce bottlenecks when the system is under heavy load, five instances of this service run concurrently, by default. Installed on all nodes but only used on the master or slave node, depending on where the IPC Core service (avid-interplay-central) is running. This service is not managed by Pacemaker, therefore you should check its status regularly, and restart it if any instance has failed. See “Verifying the AAF Generator Service” on page 77.
avid-acs-gateway	Added in MCS v2.5, this service was created to provide a common access point for the main bus services (registry, federation, attributes and infrastructure). By directing services to the gateway, connections to the bus access layer (BAL) are standardized. acs-query calls now require a local AvidGateway resource.
avid-acs-messenger	The services related to the IPC end-user messaging feature: • “messenger” service (handles delivery of user messages) • “mail” service (handles mail-forwarding feature) This service registers itself on the ACS bus. All instances are available for handling requests, which are received by way of the bus via a round-robin-type distribution system. This service operates independently, and is not managed by Pacemaker.
avid-icps-manager	Manages ICPS connections and load-balancing services.
nginx	Added in MCS v2.5, this is the web server where all client connections are processed. SSL certificate authentication are processed through nginx. Client calls are proxied to either AvidIPC (avid-interplay-central) or the avid-icps-manager for playback.

Service	Description
corosync	Cluster Engine Data Bus
pacemaker	Cluster Management and Service Failover Management
rabbitmq-server	Messaging broker/queue for the ACS bus. Maintains its own cluster functionality to deliver high-availability.
glusterd	GlusterFS daemon responsible for cache replication.
avid-mpd (if installed)	Media Distribute services. Operates similarly to the avid-acs-messenger service described above. This service is only available when Media Distribute (separate installer) is installed on the system.
"Media Index services" (if configured)	Services related to the Media Index service include: • avid-acs-search • avid-acs-autocomplete • avid-acs-media-index-configuration • avid-acs-search-import (although only active on one node) • avid-acs-media-index-feed • avid-acs-media-index-status-provider • avid-acs-media-index-permission • avid-acs-media-index-thesaurus (added in MCS v2.4) • elasticsearch • elasticsearch-tribe • pam-agent (for Interplay PAM configurations) These services are only running when Media Index has been enabled.

Cluster - Pacemaker Resources

The following table lists the cluster resources overseen and managed by Pacemaker. For additional details, query the Cluster Resource Manager using the following command:

```
crm configure show
```

In the output that appears, "primitive" is the token that defines a cluster resource.

Resource	Description
AvidAll	Encapsulates: avid-all
AvidACS	Encapsulates: avid-acs-ctrl-core
AvidAsset	Encapsulates: avid-asset
AvidAssetGc	Encapsulates: avid-asset-gc
AvidClusterMon	Encapsulates: avid-monitor

Resource	Description
AvidConnectivityMon	Encapsulates: The pingable IP address used when creating the cluster
AvidGateway	Encapsulates: avid-acs-gateway
AvidIam	Encapsulates: avid-iam
AvidICPS	Encapsulates: avid-icps-manager
AvidIPC	Encapsulates: avid-interplay-central
AvidNginx	Encapsulates: nginx
AvidServiceManager	Encapsulates: avid-acs-service-manager
AvidUMS	Encapsulates: avid-ums
AvidUpstream	Encapsulates: avid-upstream
AvidUSS	Encapsulates: avid-uss
drbd_postgres	Encapsulates: • drbd • postgresql-9.1
MongoDB	Encapsulates: mongod
Redis	Encapsulates: redis
AvidCCC	Encapsulates: avid-ccc
“Multi-Zone resources”	The following resources (and related services) are used in Multi-Zone configurations: • pgpool (pgpool) • pgpoolchecker (pgpoolchecker)
“Media Index resources”	The following resources (and related services) are used in Media Index configurations: • AvidSearch (avid-acs-search) • AvidSearchAutoComplete (avid-acs-autocomplete) • AvidSearchConfig (avid-acs-media-index-configuration) • AvidSearchImport (avid-acs-search-import) • AvidSearchIndexFeed (avid-acs-media-index-feed) • AvidSearchIndexStatus (avid-acs-media-index-status-provider) • AvidSearchPermission (avid-acs-media-index-permission) • AvidSearchThesaurus (avid-acs-media-index-thesaurus) • elasticsearch (elasticsearch) • elasticsearchTribe (elasticsearch-tribe) These resources are only active after Media Index has been configured. Except for the AvidSearchImport resource, each of these resources can be identified in the cluster with an Everywhere option such as: AvidSearchEverywhere [AvidSearch]

Interacting with Services

MCS services are standard Linux applications and/or daemons, and you interact with them following the standard Linux protocols.

To interact with services, use the standard Linux command format:

- `service <service_name> <action>`

  Standard actions include the following (some services may permit other actions):

Action	Result
status	Returns the current status of the service
stop	Stops the service
start	Starts the service
restart	Stops then restarts the service

For example, if you needed to restart the avid-ums service, the following command would be used:

```
service avid-ums restart
```

Interacting with Resources

A resource is a service or a group of services that is managed by Pacemaker. You must interact with cluster resources using the Pacemaker Cluster Resource Manager, *crm*.

To interact with resources, use the custom CRM command format:

- `crm resource <action> <resource_name>`

  For example:

```
crm resource status AvidIPC
```

Returns information similar to the following:

```
resource AvidIPC is running on: wavd-mcs01
```

The following table lists additional common commands for interacting with cluster resources:

Action	Result
status	Returns the current status of the resource. Issuing the `crm resource status` command without specifying a resource returns the status of all cluster resources (similar to what you would see in the crm_mon tool).
cleanup	Resets the fail count for the cluster resource
stop	Stops the resource and the associated service
start	Starts the resource and the associated service
restart	Stops then restarts the service

When issuing a command, the resource on the node you are logged into is affected. If interacting with an "Everywhere" resource, all nodes are affected by the command.

If you stop a resource's underlying service without going through the cluster resource manager, Pacemaker sees this as a failed service and attempts to restart it immediately. This process increases the failure count of the corresponding resource which can result in an unexpected failover. The cluster resource manager should be used in most cases when interacting with managed services.

Under special circumstances (such as during troubleshooting), you can shut down Pacemaker and Corosync, then directly stop, start and re-start the underlying services managed by Pacemaker. The simplest way to gain direct access to a node's managed services is by taking the node offline. For more information, see "Temporarily Removing a Node" on page 88.

For information on additional commands, see http://clusterlabs.org/man/pacemaker/. Be aware that some of the commands at this link can break the cluster configuration. Other commands might not apply to MCS cluster environments. Be careful when issuing cluster commands that you are unfamiliar with.

Using the avid-ics Utility Script

"avid-ics" is a utility script (not a service) that can be used to verify the status of all the major MCS services.

The script verifies the status of services such as:

- Avid services
- RabbitMQ
- sharded Mongo
- and more...

The utility script enables you to *stop*, *start* and view the *status* of all the services it encapsulates at once. Note that the utility script cannot be invoked like a true service. The form "*service avid-ics status*" will not work.

To interact with the script, use the following commands:

- **`avid-ics status`**
- **`avid-ics stop`**
- **`avid-ics start`**
- **`avid-ics help`** - Provides a list of additional commands and options.

An example output of the script will not be provided here as the results can be lengthy.

Do not use the "avid-ics start" or "avid-ics stop" commands in a cluster configuration as these commands directly affect the services and not the cluster resources. Stopping the services through avid-ics bypasses pacemaker which can lead to resource failures, errors and potential failover events.

Verifying the Startup Configuration for Avid Services

Linux includes a utility called *chkconfig* which enables a user to check the runlevels of various services. Runlevels determine the state of the service upon boot. The MCS installation process includes steps to verify or alter the runlevels of some services such as *glusterd* and *postfix*.

To run the chkconfig utility:

```
chkconfig --list
```

If desired, you can limit the output of the utility to list only services that include “avid” in the name of the service:

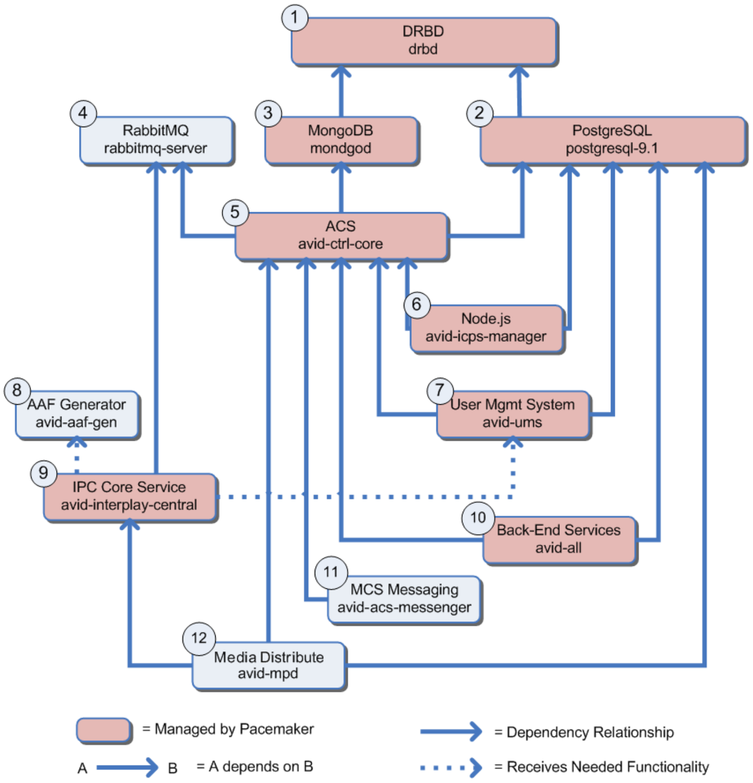
```
chkconfig --list | grep avid
```

Services Start Order and Dependencies

When direct intervention with a service is required, take special care with regards to stopping, starting, or restarting. The services on an MCS server operate within a framework of dependencies. Services must be stopped and started in a specific order. This order is particularly important when you have to restart an individual service (in comparison to rebooting the entire server). Before doing anything, identify and shut down the services that depend on the target service.

If you are running a clustered configuration, make sure to take the node offline prior to stopping any services. If you do not, Pacemaker will attempt to restart services which can result in unexpected failovers. For more information, see “Temporarily Removing a Node” on page 88.

The following illustration shows the start order and dependency relationships of some of the main cluster services. While a cluster is shown here, the theory also applies to single-server configurations.



The following table summarizes the order in which services can be safely started:

Start Order	Service Name	Process Name	Notes
1	DRBD	drbd	Only applies to cluster configurations.
2	PostgreSQL	postgresql-9.1	
3	MongoDB	mongod	
4	RabbitMQ	rabbitmq-server	
5	Avid Common Service bus (ACS: “the bus”)	avid-acs-ctrl-core	
6	Node.js	avid-icps-manager	
7	User Management Services (UMS)	avid-ums	
8	AAF Generator	avid-aaf-gen	Five instances of this service should always be running. See “Verifying the AAF Generator Service” on page 77.
9	IPC Core Services	avid-interplay-central	
10	ICPS Backend Services	avid-all	
11	ICS Messaging	avid-acs-messenger	
12	Media Distribute	avid-mpd	Only found on systems with Media Distribute installed.

Example: Restarting the User Management Services

The following example will attempt to demystify the illustration and table. Suppose you need to restart the User Management Services (avid-ums) on a single server configuration:

1. Identify its position in the dependency table (#7).
2. Identify all the services that are directly or indirectly dependent on it (service #8, #9 & #12).
3. Stop the dependent services first in order from most dependencies to least dependencies

   That is, stop service #12 first, then #9, #8, and #7.
4. Restart UMS (#7).
5. Restart services #8, #9 and, #12, in that order.

In a clustered configuration, avid-ums is managed by the AvidUMS resource and is owned by the master node. Stopping the service in a cluster could be accomplished in one of two ways:

- Complete a failover to the slave node by putting the master node into standby. This will stop all cluster resources and managed services on the master node.
- Put all nodes into standby starting with the load-balancing nodes, followed by the slave and finally the master node. This essentially stops all cluster resources and managed services.

Cluster configurations can be complex and stopping services can lead to unforeseen issues if you are not sure of what you are doing. If you believe there is an issue that requires you to stop services or resources, contact Avid Customer Care.

4 User Management

The *MediaCentral | UX Administration Guide* provides details on user creation and general user management. Appendix A of the Administration Guide provides additional information regarding commands that can be used with the avid-ums service.

This chapter includes information on determining what users are connected to the MCS system and a process for manually backing up and restoring the MCS user database.

Identifying Connected Users and Sessions

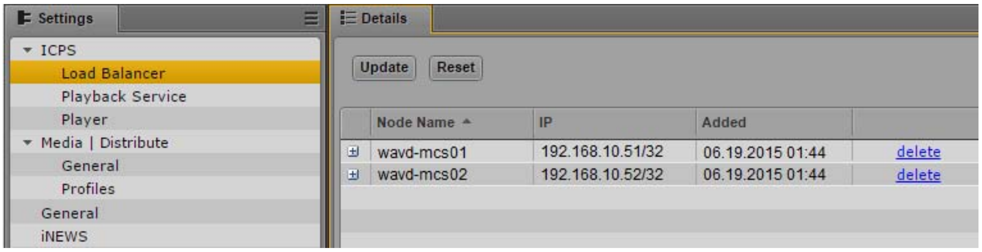
There are multiple ways to determine which users logged into MediaCentral UX, however each of the four methods below provide slightly different detail. Review each of the following options and determine which best meets your needs.

To Identify Sessions Through the MediaCentral UX System Settings Layout

MediaCentral UX provides a built-in view of connected Hosts and Session Start times based on the connected client's IP address.

1. Sign in to MediaCentral UX as a user with administrator privileges.
2. Select System Settings from the Layout menu.
3. Select MCPS > Load Balancer on the left side of the page.

   A list of all known nodes appears on the right side of the page:

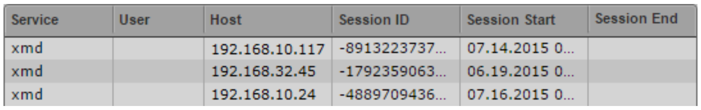


	Node Name	IP	Added	
⊞	wavd-mcs01	192.168.10.51/32	06.19.2015 01:44	delete
⊞	wavd-mcs02	192.168.10.52/32	06.19.2015 01:44	delete

   If you are running a single server configuration, only the one server will appear.

4. Click the plus sign (+) to the left of one of the nodes.

   Information regarding client connections to this node appears. Example:



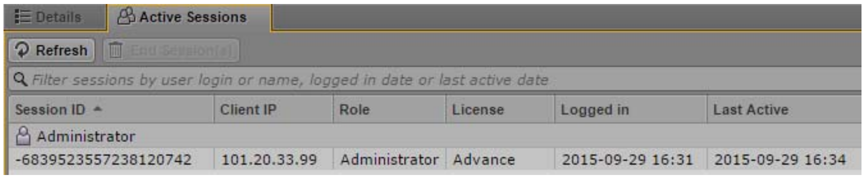
Service	User	Host	Session ID	Session Start	Session End
xmd		192.168.10.117	-8913223737...	07.14.2015 0...	
xmd		192.168.32.45	-1792359063...	06.19.2015 0...	
xmd		192.168.10.24	-4889709436...	07.16.2015 0...	

   The Host column indicates the IP address of the system that is making the connection to MediaCentral UX.

To Identify Users Through the MediaCentral UX Users Layout

The MediaCentral UX Users layout includes an Active Sessions tab which details which users are logged into the system at the current time as well as their role, license type and more.

1. Sign in to MediaCentral UX as a user with administrator privileges.
2. Select Users from the Layout menu.
3. Select the Active Sessions tab on the right-side of the interface.


Details
Active Sessions
Refresh
End Session(s)
Filter sessions by user login or name, logged in date or last active date

Session ID	Client IP	Role	License	Logged in	Last Active
Administrator					
-6839523557238120742	101.20.33.99	Administrator	Advance	2015-09-29 16:31	2015-09-29 16:34

To Identify Users and Sessions Through Logging

The *session.log* file contains much of the same information found in the Active Sessions tab of the Users layout. The benefit of the log file is that it contains a historical record of this data. The log file is located at: `/var/log/avid/avid-ums/`.

The following excerpt from the session.log file shows two separate logins:

```
2015-07-29 14:28:07.074 -0400 INFO com.avid.uls.bl.session.impl.SessionHolder - 
Logging in: logon=Administrator, role=Administrator, userId=1, 
isAvidAdministrator=true, clientIp=192.168.10.101
2015-07-29 14:28:07.075 -0400 INFO com.avid.uls.bl.session.impl.SessionHolder - 
Session created, SID=-8440723131642335013, logon=ADMINISTRATOR

2015-07-29 15:25:43.324 -0400 INFO com.avid.uls.bl.session.impl.SessionHolder - 
Logging in: logon=TestJourn, role=Journalist, userId=249, 
isAvidAdministrator=false, clientIp=192.168.10.117
2015-07-29 15:25:43.326 -0400 INFO com.avid.uls.bl.session.impl.SessionHolder - 
Session created, SID=-8917047212884686433, logon=TESTJOURN
```

For best results for viewing the log file, use an application such as Notepad++ which correctly interprets carriage returns.

To Identify Users and Sessions Through the UMS Service

The “avid-ums-statistics” command provides information about the current number of open sessions to MediaCentral UX. It also provides additional information about the total number of users and groups in the user database.

Example output of the `avid-ums-statistics` command:

```
[root@wavd-mcs01 ~]# avid-ums-statistics
Product info:
     Name:    Avid User Management Service
     Version: 2.3.0.4
Statistics:
     Amount of open sessions : 3
     Amount of users in DB   : 50
     Amount of groups in DB  : 23
     Amount of records in DB : 784
```

Backing Up the UMS Database

The *MediaCentral Platform Services Upgrade Guide* includes a process for backing up the MCS databases and system settings through the use of the *system-backup.sh* script. That process includes a backup of the UMS user database among other system settings. However, in some situations you might need to backup **only** the UMS data. For example, you may want to update the MCS database of a test system with user names and passwords, roles, and so on, from a MCS system in a production setting. This section provides the procedures for doing so.

Depending on your version of MCS, there are two different processes used to backup the user database:

- Migrating the 1.6.x (or later) UMS Database
- Migrating the 1.4.x / 1.5.x UMS Database

Migrating the 1.6.x (or later) UMS Database

To extract the UMS database from an ICS 1.6.x (or later) system, you use the avid-ums-backup and avid-ums-restore utilities located in: /opt/avid/bin

To extract the UMS database:

1. Log in to the MCS server as the *root* user.

   In a clustered configuration, log in to the master node.

2. Navigate to a location where the user database file can be created. For example:

   `cd /media`

3. Run the backup script to extract the UMS database:

   `avid-ums-backup <backup-filename> [-pp <postgres password>] [-pu <postgres user>]`

   For example:

   ```
   avid-ums-backup mydatabase -pp Avid123 -pu postgres
   ```

   The system responds with an indication of success:

   ```
   UMS database was backed up successfully.
   ```

   A new file will be created in the location where the script was run. In the example above, a single file called “mydatabase” was created in the `/media` folder.

4. Copy the backup file to an external location in preparation for restoring it to the destination MCS system.

To restore the UMS database:

1. Log in to the MCS server as the *root* user.

   In a clustered configuration, log in to the master node.

2. Stop the UMS service:

   - For a single server: **`service avid-ums stop`**
   - For a cluster: **`crm resource stop AvidUMS`**

3. Copy the backup of the UMS database to your destination MCS server.

4. Restore the UMS database:

   ```
   avid-ums-restore <backup-filename> [-pp <postgres password>] [-pu <postgres user>]
   ```

   For example:

   ```
   avid-ums-restore mydatabase -pp Avid123 -pu postgres
   ```

5. The restore script will ask you to confirm that you want to restore the database:

   ```
   Are you sure you want to perform a restore? This operation will replace the
   entire user database and remove all current users. Make sure that you have
   stopped all User Management Service instances. [Y/N]
   ```

   Once you confirm the restore request, the operation begins. Be patient as this process can take a minute or two.

   The system responds with an indication of success:

   ```
   UMS database was restore successfully.
   ```

   You may also see the following message which is normal and can be ignored:

   ```
   ************ WARNING ************
   ALTER ROLE
   ```

6. Once the user database has been restored, restart the UMS service.

   - For a single server: **`service avid-ums start`**
   - For a cluster: **`crm resource start AvidUMS`**

7. Sign in to MediaCentral UX and verify that user accounts are present and that users can log in normally.

Migrating the 1.4.x / 1.5.x UMS Database

To extract the UMS database from an ICS 1.4.x/1.5.x system and load it into an MCS 2.x system, you must use PostgreSQL tools directly, at both ends.

To extract the UMS database from an ICS 1.4.x/1.5.x system:

1. Log in to the master node as *root* and dump the UMS database

   ```
   pg_dump –U postgres uls > uls_backup.sql
   ```

2. Move the file to a safe location (off the server) in preparation for restoring it to the MCS 2.x system.

To restore the ICS 1.4.x/1.5.x UMS database to the MCS 2.x system:

1. Log in to the master node as *root*.
2. Stop the UMS service:

   - For a single server: **`service avid-ums stop`**
   - For a cluster: **`crm resource stop AvidUMS`**

3. Drop the current UMS database from the ICS database:

   ```
   psql –U postgres –c "drop database uls;"
   ```

4. Create a new UMS database:

   ```
   psql –U ulsuser postgres –c "create database uls;"
   ```

5. Import the ICS 1.5 UMS database:

   ```
   psql –U ulsuser uls < uls_backup.sql
   ```

6. Start the UMS service:
   - For a single server: **`service avid-ums start`**
   - For a cluster: **`crm resource start AvidUMS`**

5 Validating the System

This chapter includes a series of tests for determining if the underlying systems on which the MCS server is built are operating as expected. Many of the procedures in this chapter only needed to be completed once, after the initial configuration of the system. However, if conditions on the network have changed (for example, a network switch has been altered or replaced) or if you are configuring a new node for an existing cluster, these verification steps should be repeated.

For information and procedures directed towards regular maintenance activities, see "Best Practices for MediaCentral" on page 110.

Verifying the Network

This section covers the following system verification steps:

- Verifying the "Always-On" IP Address

  This cluster-specific step verifies an IP address entered during the cluster creation process.
- Verifying Network Connectivity

  Use the network "ping" command to check for other networked systems.
- Verifying Network Routes

  Use the "traceroute" command to check the number of routes between servers.
- Verifying DNS Host Name Resolution

  Poll the DNS server to verify systems are correctly registered.

When dealing with cluster configurations, recall that all nodes appear to systems outside of the cluster as a single machine with one host name and IP address. However, inter-node communication is completed using the node's individual host names and IP addresses. Additionally, in most cases, inter-cluster communication occurs over a multicast broadcast using a cluster defined multicast address. In all cases, MCS depends on reliable network connectivity for its success.

Verifying the "Always-On" IP Address

In a cluster configuration, a "pingable IP" address is assigned through the setup-corosync command. The "pingable IP" or "always-on" IP address is used by the Avid Connectivity Monitor to determine if a particular node is still in the cluster. For example, if the Connectivity Monitor on a slave node can no longer communicate with the master node, it pings the always-on IP address (in practice, usually a router). If the always-on address responds, the node concludes that the master node that has gone off-line, and assumes the role of master. If the always-on address does not respond, the slave node concludes there is a network connectivity problem and it does not attempt to take on the master role.

To obtain the pingable IP address:

On any node in the cluster type the following command:

```
crm configure show
```

This displays the contents of the Cluster Information Base in human-readable form. The pingable IP address is held by the **AvidConnectivityMon** primitive (192.168.10.1 in the example below):

```
primitive AvidConnectivityMon ocf:pacemaker:ping \
     params host_list="192.168.10.1" multiplier="100" \
     op start interval="0" timeout="20s" \
     op stop interval="0" timeout="20s" \
     op monitor interval="10s" timeout="30s"
```

Verifying Network Connectivity

Verifying basic network connectivity between the MCS server(s) and external systems such as Interplay Production or Interplay MAM servers can be an important first step in verifying system functionality. If you have a cluster configuration, verifying that all cluster node can contact each other is especially critical. The network “ping” command is a quick way to ensure good network communication.

Be aware that some network administrators block the ping command for security purposes. If ping tests fail, contact the local network admin to verify if ping is available.

To verify network connectivity:

On any network connected machine (preferably one of the cluster nodes), use the Linux *ping* command to reach the host in question:

```
ping -c # <hostname or ip address>
```

In this example `ping` is used with the `-c` switch which instructs Linux to attempt the ping at a count of # times. `<hostname or ip address>` indicates that a host name or IP address can be used.

For example:

```
ping -c 4 wavd-mcs02
```

The system responds by outputting its efforts to reach the specified host, and the results. For example, output similar to the following indicates success:

```
PING wavd-mcs02.wavd.com (192.168.10.52) 56(84) bytes of data.
64 bytes from wavd-mcs02.wavd.com (192.168.10.52): icmp_seq=1 ttl=64 time=0.086 ms
64 bytes from wavd-mcs02.wavd.com (192.168.10.52): icmp_seq=2 ttl=64 time=0.139 ms
64 bytes from wavd-mcs02.wavd.com (192.168.10.52): icmp_seq=3 ttl=64 time=0.132 ms
64 bytes from wavd-mcs02.wavd.com (192.168.10.52): icmp_seq=4 ttl=64 time=0.175 ms
```

Note the response time for each ping request. Very high response times (multiple seconds) should be investigated. This could indicate a hardware issue such as a bad transceiver or network cable.

Verify that you can ping the following:

- Host systems such as the Avid shared storage, Interplay Production Engine, iNEWS server, MAM server, etc.
- Each cluster node
- The “always on” IP address specified during the cluster configuration

Verifying Network Routes

To reduce network latency, administrators should connect the Avid servers to the network while keeping the number of “hops” to a minimum. Network “hops” refer to the number of routes, routers or network switches that data must pass through on the way from the source to the destination (and vice-versa).

The process below can be used in single-server configurations to determine the number of routes between the MCS server and other networked systems such as an Avid iNEWS server. In cluster configurations, it is especially important that there are as few network hops as possible between the clustered nodes. Ideally, there should be at most one hop.

To verify routing between networked systems:

Issue the following command from any MCS server:

```
traceroute <hostname>
```

For example, issuing a traceroute on “localhost” (always your current machine) will result in output similar to the following, representing a single “hop”:

```
traceroute to localhost (127.0.0.1), 30 hops max, 60 byte packets
  1 localhost (127.0.0.1) 0.020 ms 0.003 ms 0.003 ms
```

For a machine that is three network hops away, the results will resemble the following:

```
traceroute to wavd-mc11 (192.168.32.11), 30 hops max, 60 byte packets
1  192.169.18.1 (192.168.18.1) 0.431 ms 0.423 ms 0.416 ms
2  gw.wavd.com (192.168.32.7) 0.275ms 0.428 ms 0.619 ms
3  192.168.48.40 (192.168.48.40) 0.215 ms 0.228 ms 0.225 ms
```

Repeat the traceroute tests as needed. In cluster configurations, each node should have the same number of “hops”. If one or more nodes has a different number of hops than the others, this should be investigated and optimized if possible.

Be sure to run traceroute on the cluster’s pingable IP address to verify it is within easy reach and is unlikely to be made unreachable, for example, by inadvertent changes to network topology.

Verifying DNS Host Name Resolution

It is important that the Domain Name System (DNS) servers correctly identify all Avid servers. This is true of all physical servers, cluster nodes, and all virtual cluster IP and hostnames. The Linux *dig* (domain information groper) and *nslookup* commands perform similar name lookup functions.

Enter the following commands as the *root* user.

Using “dig” to verify DNS:

```
dig +search <host>
```

The *+search* option forces *dig* to use the DNS servers defined in the `/etc/resolve.conf` file, in the order they are listed in the file.

The *dig* command as presented above returns information on the “A” record for the host name submitted with the query, for example:

```
dig +search wavd-mcs01
```

Returns output similar to the following:

```
[root@wavd-mcs01 ~]# dig +search wavd-mcs01
; <<>> DiG 9.8.2rc1-RedHat-9.8.2-0.17.rc1.el6_4.6 <<>> +search wavd-mcs01
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 63418
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;wavd-mcs01.wavd.com.                IN      A

;; ANSWER SECTION:
wavd-mcs01.wavd.com. 3600    IN      A       192.168.10.51

;; Query time: 0 msec
;; SERVER: 192.168.10.10#53(192.168.10.10)
;; WHEN: Tue Jul 4 15:57:25 2015
;; MSG SIZE  rcvd: 56
```

Even though the command specified the short hostname, the “ANSWER SECTION” provides the Fully Qualified Domain Name (FQDN) as well as the IP address of 192.168.10.51.

Additionally, the “>>HEADER<<” section indicated a status of **NOERROR**. This verifies that the DNS server (192.168.10.10 in this example) has a valid entry for the host in question. The following table presents other possible return codes:

Return Code	Description
NOERROR	DNS Query completed successfully
FORMERR	DNS Query Format Error
SERVFAIL	Server failed to complete the DNS request
NXDOMAIN	Domain name does not exist
NOTIMP	Function not implemented
REFUSED	The server refused to answer for the query
YXDOMAIN	Name that should not exist, does exist
XRRSET	RRset that should not exist, does exist
NOTAUTH	Server not authoritative for the zone
NOTZONE	Name not in zone

Using “nslookup” to verify DNS:

```
nslookup <host> or <ip>
```

nslookup polls the primary DNS server configured in the `resolv.conf` file for the hostname or IP address you specify. For example:

```
nslookup wavd-mcs01
```

Returns output similar to the following:

```
Server:         192.168.10.10
Address:        192.168.10.10#53

Name:   wavd-mcs01.wavd.com
Address: 192.168.10.51
```

Note that DNS servers contain both forward and reverse zones. By entering a hostname in the nslookup command, only the forward zone information was verified. Repeat the command using the IP address to verify the reverse zone.

Validating the FQDN for External Access

It is vital that the fully qualified domain name (FQDN) for each MCS server is resolvable by the domain name server (DNS) tasked with doing so. This is particularly important when MCS is accessed from the MediaCentral mobile application (iPad, iPhone or Android device) or when connecting from outside the corporate firewall through Network Address Translation (NAT). In such cases, review the FQDN returned by the XLB load-balancer. Ensure that the network administrator has assigned the FQDN a unique public IP address.

Currently, connecting to MCS through NAT is only supported for single-server configurations and not MCS cluster configurations.

Verifying External Access

1. Launch a web browser on your client(s) of interest. This could be:
   - An iPad, iPhone or Android device
   - A client outside of the corporate firewall through a VPN or NAT connection
   - A client within the corporate firewall
2. Enter the following URL into the address bar:

   http://<*FQDN*>/api/xlb/nodes/less/?service=xmd

   Where <*FQDN*> is the fully qualified domain name of the MCS server. In a cluster configuration, enter the FQDN of the cluster (virtual cluster hostname). For example:

   http://wavd-mcs.wavd.com/api/xlb/nodes/less/?service=xmd

The system returns a string similar to the following (line breaks added for clarity):

```
{"status":"ok","data":
{"xlb_service_ip":"10.XXX.XXX.XX",
"xlb_service_port":5000,
"xlb_node_ip":"10.XXX.XXX.XX/32",
"xlb_node_name":"wavd-mcs01",
"xlb_node_full_name":"wavd-mcs01.subdomain.domain.net"}}
```

Note the following data of interest:

Item	Description
xlb_node_ip	The IP address of the node assigned to you for the current session. In a cluster configuration, this will be one of the cluster nodes.
xlb_node_name	The host name of the node assigned to you for the current session. In a cluster configuration, this will be one of the cluster nodes.
xlb_node_full_name	The FQDN of the assigned node. If connecting to MediaCentral from outside the corporate firewall through NAT, this domain name must resolve to an external (public) IP address.

An example of a failed connection from the Safari browser on an iOS device appears as follows: "Safari cannot open the page because the server cannot be found."

3. Verify the output of the command.

   For a Single Server:

   In a single server configuration, the "xlb_node_full_name" should match the FQDN name entered in the Server field of the MediaCentral System Setting (System Settings > MPCS > Player > server).

   For a Cluster:

   In a cluster configuration, the domain extension (e.g. wavd.com) displayed in "xlb_node_full_name" should match the domain extension used in the Server field of the MediaCentral System Setting (System Settings > MCPS > Player > Server).

   In this case you are only matching the domain extension because the Server field in the MediaCentral System Settings specified the cluster name and not an individual node.

   The "xlb_node_full_name" will not return the cluster FQDN, but will instead return one of the cluster's individual node names. The returned node name is based on whichever node is most available to respond for the current session.

Refreshing the web page may return a different node name. This is normal.

If the output does not match, you may be able to sign in to MediaCentral UX on a remote client, but playback may not function.

If MediaCentral UX will be accessed from outside the corporate firewall through NAT, ensure that this server is accessible. In particular, ensure the FQDN returned by the query is associated with a public address.

Troubleshooting

If you are not getting the results you expect, work with your on-site IT Department to verify that your DNS includes forward and reverse entries for each MCS server and an entry for the virtual cluster hostname and IP. Make sure there are no duplicate entries that contain incorrect information (e.g. an invalid IP address).

If you are still unsuccessful and you are not using NAT, an alternative option exists. MCS v2.0.2 added a feature for altering the "application.properties" file to instruct the MCS servers to return an IP address during the load-balancing handshake instead of a hostname.

This process is not supported for single-server systems using NAT.

To adjust the application.preperties file:

1. Log in to the MCS server as the 'root' user. If you have a clustered configuration, log into the master node.
2. Navigate to the following directory:

   ```
   cd /opt/avid/etc/avid/avid-interplay-central/config
   ```

3. This directory contains an "application.properties.example" file. The example file includes information on some features that can be adjusted. Use the following command to rename this file to exclude the ".example" extension:

   ```
   mv application.properties.example application.properties
   ```

4. Edit the file using a text editor (such as vi):

   ```
   vi application.properties
   ```

5. Add the following text to the end of the file:

   ```
   system.com.avid.central.services.morpheus.media.UseIpForPreferredHost=true
   ```

6. Save and exit the vi session. Press <ESC> and type: `:wq`
7. Repeat steps 1 – 6 on the slave node.
8. Once complete, the AvidIPC resource must be restarted.

   This step will disconnect any users currently working on the system.

   a. If running a single server configuration, issue the following command:

      ```
      service avid-interplay-central restart
      ```

   b. If running a clustered configuration, issue the following command on any node in the cluster:

      ```
      crm resource restart AvidIPC
      ```

9. Once this process is complete, repeat the process for validating the FQDN of the MCS Servers.

Verifying Time Synchronization

Verifying time synchronization across multiple networked servers in Linux is a challenge, and there is no simple way to do it that provides entirely satisfactory results. The major impediment is the nature of the Linux Network Time Protocol (NTP) itself. If the MCS servers are not in time sync with external systems such as the Media Indexer for Interplay Production workflows, users could see Media Offline messages in the Media pane. In clustered configurations, Pacemaker and Corosync rely on time stamps for accurate communication. Failure to synchronize clustered nodes could result in unexpected failover events.

During MCS installation, a *cron* job was created to synchronize each MCS server to an NTP time server. Note that the time adjustment is not instantaneous — it can take some time for the NTPD daemon to adjust the local system time to the value retrieved from the NTP time server. Furthermore, network congestion can result in unpredictable delays between each server seeking accurate time, and accurate time being returned to it.

For all of these the reasons, there is no guarantee that all systems will see the same time at the same moment. Nevertheless, some basic verification steps can be performed:

- Verify the NTP configuration file (`/etc/ntp.conf`) contains the address of an in-house NTP server.
- Ensure any out-of-house servers (e.g. “`0.rhel.pool.ntp.org`”) are commented out or removed from the `ntp.conf` file.
- Verify the NTP server in the NTP configuration file is reachable from each server in a cluster:

  ```
  ntpdate -q <server_address>
  ```
- Verify a *cron* job (`/etc/cron.d/ntpd`) has been created.
- In a cluster, create multiple SSH (PuTTY) sessions (one to each server) and visually verify the system date, time and timezone using the **`date`** command.
- If needed, use NTP to adjust the time and date:

  ```
  /usr/sbin/ntpd -q -u ntp:ntp
  ```

Some industry literature suggests a server's time can take some time to “settle down” after a reboot, or after requesting a clock synchronization using NTP. It is not unusual for there to be delays of up to an hour or two before clock accuracy is established.

For more information see “Configure Date and Time Settings” in the *Avid MediaCentral Platform Services Installation and Configuration Guide*.

For additional information on time synchronization, see Time Synchronization for Avid Interplay Systems on the Avid Knowledge Base.

Verifying ACS Bus Functionality

The Avid Common Services bus (“the bus”) provides essential bus services needed for the overall platform to work. Numerous services depend upon it, and will not start — or will throw serious errors — if the bus is not running. You can easily verify ACS bus functionality using the acs-query command. In a cluster configuration, issuing this command on the master node tests the ACS bus directly. Although the ACS bus operates on the master and slave nodes only, running acs-query on a non-master node validates network and node-to-node bus connectivity.

To verify the connection to the ACS bus:

Query the ACS bus database using the acs-query command with using the --path option:

```
acs-query --path=serviceType
```

Output similar to the following should be returned:

```
"avid.acs.registy"
```

The above output indicates RabbitMQ, MongoDB and PostgreSQL are all running and reachable by the ACS bus (since no errors are present). It also indicates the “avid.acs.registry” bus service is available.

Verifying the Status of RabbitMQ

RabbitMQ is a messaging bus used by the top-level MCS services to process requests between connected systems. In a cluster configuration, RabbitMQ maintains its own cluster functionality independent of the Corosync cluster.

To verify that RabbitMQ is functioning properly:

Request the status of the messaging bus using the “rabbitmqctl” command:

```
rabbitmqctl cluster_status
```

Example output of a single-server configuration:

```
[root@wavd-doc01 ~]# rabbitmqctl cluster_status
Cluster status of node 'rabbit@wavd-doc01' ...
[{nodes,[{disc,['rabbit@wavd-doc01']}]},
  {running_nodes,['rabbit@wavd-doc01']},
  {cluster_name,<<"rabbit@wavd-doc01">>},
  {partitions,[]}]
```

Example output for a two node cluster:

```
[root@wavd-mcs01 ~]# rabbitmqctl cluster_status
Cluster status of node 'rabbit@wavd-mcs01' ...
[{nodes,[{disc,['rabbit@wavd-mcs01','rabbit@wavd-mcs02']}]},
  {running_nodes,['rabbit@wavd-mcs01','rabbit@wavd-mcs02']},
  {cluster_name,<<"rabbit@wavd-mcs01.wavd.com">>},
  {partitions,[]}]
  ...done.
```

If you do not see similar results or need additional information on RabbitMQ, including troubleshooting assistance, see:

http://avid.force.com/pkb/articles/en_US/troubleshooting/RabbitMQ-cluster-troubleshooting

Verifying the AAF Generator Service

The AAF Generator service (*avid-aaf-gen*) is responsible for saving sequences. To reduce the possibility of bottlenecks when many users attempt to save sequences at the same time, multiple instances of the service run simultaneously (by default, five). As a result, MCS has the ability to save multiple sequences concurrently, significantly reducing overall wait-times under heavy load.

In a cluster deployment, this service is installed and running on all nodes. However, it is only involved in saving sequences on the master node.

The avid-aaf-gen service is not managed by Pacemaker. Therefore, it is important to regularly verify its status manually. If one or more instances of the service has failed, it should be restarted. An instance can fail, for example, if an invalid AAF is used within a sequence. If all instances of the avid-aaf-gen service fail, the IPC core service (avid-interplay-central), assumes the responsibility for saving transfers and bottlenecks can arise.

Logs related to this service are created at: `/var/log/avid/avid-aaf-gen/log_xxx`.

To verify the status and/or stop the AAF Generator service:

1. Log in to the MCS server. If on a cluster log in to both the master and slave nodes as root.

   Although the AAF Generator service is active in saving sequences only on the master node, you should also verify its status on the slave node, to prepare for any failover.

2. Verify the status of the AAF Generator service:

   ```
   service avid-aaf-gen status
   ```

   The system outputs the status of each instance, similar to the following:

   ```
   avid-aaf-gen_1 process is running                    [  OK  ]
   avid-aaf-gen_2 process is running                    [  OK  ]
   avid-aaf-gen_3 process is running                    [  OK  ]
   avid-aaf-gen_4 process is running                    [  OK  ]
   avid-aaf-gen_5 process is running                    [  OK  ]
   ```

   An error would look like this:

   ```
   avid-aaf-gen_1 process is not running                [WARNING]
   ```

3. In the event of an error, restart the service as follows:

   ```
   service avid-aaf-gen restart
   ```

   Output similar to the following indicates the service has restarted correctly:

   ```
   Starting process avid-aaf-gen_1 - Stat: 0            [  OK  ]
   Starting process avid-aaf-gen_2 - Stat: 0            [  OK  ]
   Starting process avid-aaf-gen_3 - Stat: 0            [  OK  ]
   Starting process avid-aaf-gen_4 - Stat: 0            [  OK  ]
   Starting process avid-aaf-gen_5 - Stat: 0            [  OK  ]
   ```

4. If you need to stop the service this must be done in two steps:

   a. Configure 0 instances of the service (there are 5 by default):

      ```
      echo 0 > /opt/avid/avid-aaf-gen/DEFAULT_NUM_PROCESSES
      ```

   b. With zero instances configured, you can stop the service normally:

      ```
      service avid-aaf-gen stop
      ```

5. To restart the service, reset the number of instances to the default (5) and restart the service.

Verifying Custer-Specific Components

The following topics are cluster-specific and do not apply to single-server configurations.

Verifying the DRBD Status

Recall that DRBD is responsible for mirroring the MCS database on the two servers in the master/slave configuration. It does not run on any other nodes. In this section you run the DRDB *drdb-overview* utility to ensure there is connectivity between the two DRBD nodes, and to verify database replication is taking place.

To view the status of DRBD, log in to the node of interest and issue the following command:

```
drbd-overview
```

A healthy master node will produce output similar to the following:

```
1:r0/0 Connected Primary/Secondary UpToDate/UpToDate C r----- /mnt/drbd ext4
20G 907M 18G 5%
```

A healthy slave node will return the following:

```
1:r0/0 Connected Secondary/Primary UpToDate/UpToDate C r-----
```

If the master and slave nodes do not resemble the above output, see "Troubleshooting DRBD" on page 128.

Element	Description
`1:r0/0`	The DRBD device number ("1") and name ("r0/0").
Connected	The connection state. Possible states include: • Connected - Connection established and data mirroring is active. • Standalone - No DRBD network connection (i.e., not yet connected, explicitly disconnected, or connection dropped). In MCS this usually indicates a "split brain" has occurred. • WFConnection - The node is waiting for the peer node to become visible on the network.
`Primary/Secondary`	The roles for the local and peer (remote) DRBD resources. The local role is always presented first (i.e. local/peer). • Primary - The active resource. • Secondary - The resource that receives updates from its peer (the primary). • Unknown - The resource's role is currently not known. This status is only ever displayed for the peer resource (i.e. Primary/Unknown).

Element	Description
`UptoDate/UptoDate`	The resource's disk state. The local disk state is presented first (i.e. local/peer). Possible states include: • UptoDate - Consistent and up to date. The normal state. • Consistent - Data is consistent, but the node is not connected to its peer. • Inconsistent - Data is not consistent. This occurs on both nodes prior to first (full) sync, and on the synchronization target during synchronization. • Unknown - No connection to peer. This status is only ever displayed for the peer resource (i.e. UptoDate/Unknown).
`C`	The replication protocol. Should be "C" (synchronous).
`r-----`	I/O flags. The first entry should be "r" (running).
`/mnt/drbd ext4 20G 907M 18G 5%`	The DRBD partition mount point and other standard Linux file system information. This indicates the DRBD partition is mounted on this node. This should be the case on the master node only.

Verifying the Pacemaker / Corosync Cluster Status

For all important events, such as a master node failover, the cluster sends automated e-mails to cluster administrator e-mail address(es). It is nevertheless important to regularly check up on the cluster manually. Recall that cluster resources are Linux services under management by Pacemaker. By regularly checking the fail-counts of cluster resources, for example, you can identify issues before a failover actually takes place.

For more information on the Cluster Resource Monitor, reference "Cluster Resource Monitor" on page 80.

6 Cluster Resource Monitor

The easiest way to verify that all nodes are participating in the cluster and that all resources are up is through the Pacemaker Cluster Resource Monitor, *crm_mon*. This utility provides a real-time view of the cluster status including information on failures and fail-counts. This section provides information to assist in interpreting the output of the Cluster Resource Monitor and is not applicable to single-server installations.

Accessing the Cluster Resource Monitor

To monitor the status of the cluster, log in to any node in the cluster as *root* and enter the following command:

```
crm_mon [-f]
```

The output of this command presents the status of the main resources (and underlying services) controlled by Pacemaker, and the nodes on which they are running. The optional **`-f`** switch adds fail-count information to the output.

Press CTRL-C on a Windows keyboard or CMD-C on a Mac keyboard to exit the crm_mon console.

Interpreting the Output of CRM

Line-by-Line Breakdown

The following is an example of a four-node cluster. This section provides a line-by-line explanation of typical `crm_mon` output (line numbers have been added, for reference). The output of this command varies depending on your version of MCS.

The "lsb" prefix shown in the Cluster Resource Monitor indicates the named service conforms to the Linux Standard Base (LSB) project, meaning these services support standard Linux commands for scripts (e.g. start, stop, restart, force-reload, status).
The "ocf" prefix indicates the named entity is a cluster resource, compliant with the Open Cluster Framework (OCF). OCF can be understood as an extension of LSB for the purposes of clustering.

```
1)  ============
2)  Last updated: Thu Jul 16 16:20:01 2015
3)  Last change: Mon Jul 13 10:06:51 2015 via crm_attribute on wavd-mcs02
4)  Stack: classic openais (with plugin)
5)  Current DC: wavd-mcs04 - partition with quorum
6)  Version: 1.1.11-97629de
7)  4 Nodes configured, 4 expected votes
8)  40 Resources configured
9)  ============

10)  Online: [ wavd-mcs01 wavd-mcs02 wavd-mcs03 wavd-mcs04 ]

11)   Clone Set: AvidConnectivityMonEverywhere [AvidConnectivityMon]
```

```
12)       Started: [ wavd-mcs01 wavd-mcs02 wavd-mcs03 wavd-mcs04 ]
13)  AvidClusterMon  (lsb:avid-monitor):     Started wavd-mcs01
14)  MongoDB (lsb:mongod):   Started wavd-mcs01
15)  Redis   (ocf::avid:redis):      Started wavd-mcs01
16)   Resource Group: postgres
17)       postgres_fs        (ocf::heartbeat:Filesystem):    Started wavd-mcs01
18)       AvidClusterIP      (ocf::heartbeat:IPaddr2):       Started wavd-mcs01
19)       pgsqlDB    (ocf::avid:pgsql_Avid): Started wavd-mcs01
20)   Master/Slave Set: ms_drbd_postgres [drbd_postgres]
21)       Masters: [ wavd-mcs01 ]
22)       Slaves: [ wavd-mcs02 ]
23)   Clone Set: AvidAllEverywhere [AvidAll]
24)       Started: [ wavd-mcs01 wavd-mcs02 wavd-mcs03 wavd-mcs04 ]
25)  AvidIPC (lsb:avid-interplay-central):   Started wavd-mcs01
26)  AvidUpstream    (lsb:avid-upstream):    Started wavd-mcs01
27)   Clone Set: AvidIamEverywhere [AvidIam]
28)      Started: [ wavd-mcs01 wavd-mcs02]
29)  Clone Set: AvidAssetEverywhere [AvidAsset]
30)      Started: [ wavd-mcs01 wavd-mcs02 ]
31)  Clone Set: AvidAssetGcEverywhere [AvidAssetGc]
32)      Started: [ wavd-mcs01 wavd-mcs02 ]
33)  AvidUMS (lsb:avid-ums): Started wavd-mcs01
34)  AvidUSS (lsb:avid-uss): Started wavd-mcs01
35)  AvidACS (lsb:avid-acs-ctrl-core):       Started wavd-mcs01
36)  AvidServiceManager      (lsb:avid-acs-service-manager): Started wavd-mcs01
37)   Clone Set: AvidGatewayEverywhere [AvidGateway]
38)       Started: [ wavd-mcs01 wavd-mcs02 wavd-mcs03 wavd-mcs04 ]
39)   Clone Set: AvidICPSEverywhere [AvidICPS]
40)       Started: [ wavd-mcs01 wavd-mcs02 wavd-mcs03 wavd-mcs04 ]
41)   Clone Set: AvidNginxEverywhere [AvidNginx]
42)       Started: [ wavd-mcs01 wavd-mcs02 wavd-mcs03 wavd-mcs04 ]
```

Line(s)	Description
1-9	Header information. The look of this information varies between versions of MCS.
2	Last time something changed in the cluster status (for example, a service stopped, was restarted, and so on).
3	Last time the cluster configuration was changed, and from where it was changed.
4	Name of the Corosync stack (includes Pacemaker and Corosync). Always named “openais”.
5	Displays the current holder of the configuration. If you change something on a machine, the change must be “approved” by the Current DC.
6	Version number of the Corosync stack.
7	The number of nodes configured. Expected votes relates to quorums (unused).
8	The total number of Pacemaker managed resources (services and groups of services).
10	Lists the cluster nodes including their current status (online, offline, standby).
11-12	The AvidConnectivityMon resource monitors the pingable IP address specified during the cluster setup.
13	The resource that sends the automated e-mails.
14	The MongoDB resource.
15	The Redis resource.

Line(s)	Description
16-19	The PostgreSQL resource group. · postgres_fs: Responsible for mounting the drbd device as a file system. · AvidClusterIP: The virtual cluster IP address. · pgsqlDB: The PostgreSQL database.
20-22	The master/slave set for DRBD.
23-24	The playback services. “Clone Set” indicates it is running on all nodes in the cluster.
25	The Interplay Central resource.
26	Introduced in MCS v2.6, the AvidUpstream resource runs on the master node.
27-28	Introduced in MCS v2.6, the AvidIam service runs on the master and slave nodes.
29-30	Introduced in MCS v2.9, the AvidAsset service runs on the master and slave nodes.
31-32	Introduced in MCS v2.9, the AvidAssetGc service runs on the master and slave nodes.
33	The User Management Service resource.
34	The User Setting Service resource.
35	The Avid Common Services bus (“the bus”).
36	In MCS v2.5, the Avid Service Manager resource ran on all nodes under the resource name “AvidServiceManagerEverywhere”. In MCS v2.6, the resource was altered to run on the master node only.
37-38	Introduced in MCS v2.5, the AvidGateway resource runs on all cluster nodes.
39-40	The Avid Interplay Central Playback Services (the “back end” services).
41-42	Introduced in MCS v2.5, the AvidNginx resource runs on all cluster nodes.
-- --	Media Index introduces multiple additional resources. (not shown)

Notice that while all services are running on one node — *wavd-mcs01*, in the sample output — only some of the services are running on the others. This is because *wavd-mcs01* is the master node. The slave node (*wavd-mcs02*) runs a subset of these services, many in a standby mode in the event that the master experiences a failure. *wavd-mcs03* and *wavd-mcs04* are load-balancing nodes and primarily run video playback services.

Identifying the Master, Slave and Load-Balancing Nodes

The header information at the beginning of the crm_mon tool lists the total number of nodes configured. Four nodes are listed in the example above. The “Online” section just below the header information lists which nodes are in the cluster and online. If any nodes are powered-on, but not active, they will be listed in the same section as “standby”. If any nodes are known powered-off, they will be listed as “offline”.

The master node can be identified in a number of ways:

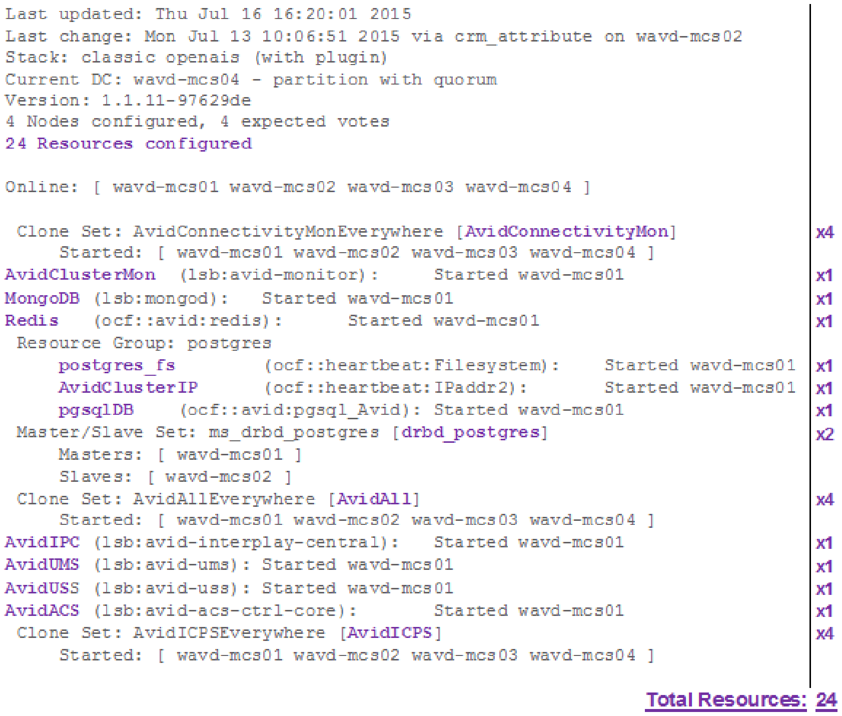
- It is always the owner of the AvidClusterIP resource.
- It is listed as “master” under the drbd_postgres resource.
- It will be the owner of multiple other resources such as: MongoDB, AvidIPC, AvidUMS and more.

The slave node can be identified as “slave” under the drbd_postgres resource. It will also run additional load-balancing resources such as AvidICPS and AvidAll.

The load-balancing nodes will only run load-balancing resources such as AvidICPS and AvidAll.

Identifying the Cluster Resources

The following image identifies the Pacemaker resources within the cluster. Your cluster may have additional resources based on how the system has been configured. For instance, Media Index configurations will have many more resources. Older versions of MediaCentral may have fewer resources configured.



```
Last updated: Thu Jul 16 16:20:01 2015
Last change: Mon Jul 13 10:06:51 2015 via crm_attribute on wavd-mcs02
Stack: classic openais (with plugin)
Current DC: wavd-mcs04 - partition with quorum
Version: 1.1.11-97629de
4 Nodes configured, 4 expected votes
24 Resources configured

Online: [ wavd-mcs01 wavd-mcs02 wavd-mcs03 wavd-mcs04 ]

 Clone Set: AvidConnectivityMonEverywhere [AvidConnectivityMon]             x4
     Started: [ wavd-mcs01 wavd-mcs02 wavd-mcs03 wavd-mcs04 ]
AvidClusterMon  (lsb:avid-monitor):     Started wavd-mcs01                  x1
MongoDB (lsb:mongod):   Started wavd-mcs01                                  x1
Redis   (ocf::avid:redis):      Started wavd-mcs01                          x1
 Resource Group: postgres
     postgres_fs        (ocf::heartbeat:Filesystem):    Started wavd-mcs01  x1
     AvidClusterIP      (ocf::heartbeat:IPaddr2):       Started wavd-mcs01  x1
     pgsqlDB    (ocf::avid:pgsql_Avid): Started wavd-mcs01                  x1
 Master/Slave Set: ms_drbd_postgres [drbd_postgres]                         x2
     Masters: [ wavd-mcs01 ]
     Slaves: [ wavd-mcs02 ]
 Clone Set: AvidAllEverywhere [AvidAll]                                     x4
     Started: [ wavd-mcs01 wavd-mcs02 wavd-mcs03 wavd-mcs04 ]
AvidIPC (lsb:avid-interplay-central):   Started wavd-mcs01                  x1
AvidUMS (lsb:avid-ums): Started wavd-mcs01                                  x1
AvidUSS (lsb:avid-uss): Started wavd-mcs01                                  x1
AvidACS (lsb:avid-acs-ctrl-core):       Started wavd-mcs01                  x1
 Clone Set: AvidICPSEverywhere [AvidICPS]                                   x4
     Started: [ wavd-mcs01 wavd-mcs02 wavd-mcs03 wavd-mcs04 ]
```

Note the total number of “Resources configured” at the top of the tool. There are 24 resources in the example image. The resources are identified in bold text and a count has been added on the right. Some resources run on the master node only while other resources, such as AvidICPS, run on multiple nodes. The counts listed on the right equal the total number of configured resources. If you are using an SSH client (PuTTY) to monitor the cluster and you do not see all the resources in the Cluster Resource Monitor, you may need to expend the size of your SSH window to see all resources on screen.

Identifying Failures in CRM

When using the **-f** switch with the `crm_mon` command, additional information regarding failures and fail-counts appear at the bottom of the monitor. During operation of the Cluster, services may fail. In some cases this is normal and expected behavior. Pacemaker automatically restarts the service and users receive no indication that a failure occurred. In other cases, a failure could represent a problem and further investigation is required. In either case, failures should not be allowed to continue unchecked as too many failures could eventually initiate a failover event. The following example uses the `crm_mon -f` command to display additional information on failures in this four-node cluster.

Alternatively, the command “crm_mon -f1” can be used. The added “1” tells the command to print the output to the screen only once and then return to the Linux command prompt.

```
Last updated: Thu Jul 16 16:20:01 2015

Last change: Mon Jul 13 10:06:51 2015 via crm_attribute on wavd-mcs02
Stack: classic openais (with plugin)
Current DC: wavd-mcs04 - partition with quorum
Version: 1.1.11-97629de
4 Nodes configured, 4 expected votes
40 Resources configured

Online: [ wavd-mcs01 wavd-mcs02 wavd-mcs03 wavd-mcs04 ]

 Clone Set: AvidConnectivityMonEverywhere [AvidConnectivityMon]
     Started: [ wavd-mcs01 wavd-mcs02 wavd-mcs03 wavd-mcs04 ]
AvidClusterMon  (lsb:avid-monitor):     Started wavd-mcs01
MongoDB (lsb:mongod):   Started wavd-mcs01
Redis   (ocf::avid:redis):      Started wavd-mcs01
 Resource Group: postgres
     postgres_fs        (ocf::heartbeat:Filesystem):    Started wavd-mcs01
     AvidClusterIP      (ocf::heartbeat:IPaddr2):       Started wavd-mcs01
     pgsqlDB    (ocf::avid:pgsql_Avid): Started wavd-mcs01
 Master/Slave Set: ms_drbd_postgres [drbd_postgres]
     Masters: [ wavd-mcs01 ]
     Slaves: [ wavd-mcs02 ]
 Clone Set: AvidAllEverywhere [AvidAll]
     Started: [ wavd-mcs01 wavd-mcs02 wavd-mcs03 wavd-mcs04 ]
AvidIPC (lsb:avid-interplay-central):   Started wavd-mcs01
AvidUpstream    (lsb:avid-upstream):    Started wavd-mcs01
Clone Set: AvidIamEverywhere [AvidIam]
     Started: [ wavd-mcs01 wavd-mcs02 ]
 Clone Set: AvidAssetEverywhere [AvidAsset]
     Started: [ wavd-mcs01 wavd-mcs02 ]
 Clone Set: AvidAssetGcEverywhere [AvidAssetGc]
     Started: [ wavd-mcs01 wavd-mcs02 ]
AvidUMS (lsb:avid-ums): Started wavd-mcs01
AvidUSS (lsb:avid-uss): Started wavd-mcs01
AvidACS (lsb:avid-acs-ctrl-core):       Started wavd-mcs01
AvidServiceManager (lsb:avid-acs-service-manager): Started wavd-mcs01
  Clone Set: AvidGatewayEverywhere [AvidGateway]
     Started: [ wavd-mcs01 wavd-mcs02 wavd-mcs03 wavd-mcs04 ]
  Clone Set: AvidICPSEverywhere [AvidICPS]
     Started: [ wavd-mcs01 wavd-mcs02 wavd-mcs03 wavd-mcs04 ]
  Clone Set: AvidNginxEverywhere [AvidNginx]
     Started: [ wavd-mcs01 wavd-mcs02 wavd-mcs03 wavd-mcs04

Migration summary:
* Node wavd-mcs01:
   Redis: migration-threshold=20 fail-count=5 last-failure='Wed Jul 15 16:46:45 2015'
   AvidUMS: migration-threshold=20 fail-count=3 last-failure='Wed Jul 15 15:26:30 2015'
   AvidACS: migration-threshold=20 fail-count=1 last-failure='Wed Jul 15 18:30:08 2015'
```

```
* Node wavd-mcs02:
   AvidConnectivityMon: migration-threshold=1000000 fail-count=1 last-failure='Wed Jul 15
18:30:49 2015'
* Node wavd-mcs03:
   AvidConnectivityMon: migration-threshold=1000000 fail-count=1 last-failure='Wed Jul 15
18:30:08 2015'
* Node wavd-mcs04:

Failed actions:
    Redis_monitor_15000 on wavd-mcs01 'not running' (7): call=5381, status=complete, last-
rc-change='Wed Jul 15 16:46:45 2015', queued=0ms, exec=0ms
    AvidUMS_monitor_25000 on wavd-mcs01 'unknown error' (1): call=5317, status=Timed Out,
last-rc-change='Wed Jul 15 15:26:30 2015', queued=0ms, exec=0ms
    AvidACS_monitor_25000 on wavd-mcs01 'unknown error' (1): call=5405, status=Timed Out,
last-rc-change='Wed Jul 15 18:30:48 2015', queued=0ms, exec=0ms
     AvidConnectivityMon_monitor_10000 on wavd-mcs02 'unknown error' (1): call=325,
status=Timed Out, last-rc-change='Wed Jul 15 18:30:49 2015', queued=0ms, exec=0ms
     AvidConnectivityMon_monitor_10000 on wavd-mcs03 'unknown error' (1): call=3216,
status=Timed Out, last-rc-change='Wed Jul 15 18:30:48 2015', queued=0ms, exec=0ms
```

The “Failed actions” area is present in the `crm_mon` tool with or without the **`-f`** option. This information has not been present in previous examples as this is the first example with failures. In this example, failures occurred on *wavd-mcs01, wavd-mcs02 and wavd-mcs03*, but no errors occurred on *wavd-mcs04*. Additionally, all services have recovered and are now running normally. A failure in the middle of the tool represents a hard failure - the resource failed and has not recovered. Failures at the end of the tool, are historical counts and do not necessarily represent a current condition.

The “Migration summary” area has been added with the use of the **`-f`** switch. It lists similar information to the “Failed actions” area: which node(s) encountered a failure, the name of the failed resource and the date/time stamp of the last failure. Additionally, this area lists the failure count. This is important information as it may not only indicate the severity of the issue, but also indicate how close the count is to the “migration-threshold” (failover).

Recall that some failures are considered normal and high fail-counts may not be a concern. As an example, the migration-threshold of the AvidConnectivityMon is 1,000,000 which is the equivalent to “infinite”. Other resources have a migration-threshold as low as 2. A failure indicates that the verification of the resource was unavailable at the requested time. This could happen for a number of reasons and may not indicate a true failure, only that the resource could not be contacted.

Failures at the bottom of the tool can be cleared using the following command in a second terminal window (a terminal window other than the one showing crm_mon):

`crm resource cleanup <`*`rsc`*`> [<`*`node`*`>]`

- <*rsc*> is the resource name of interest: AvidIPC, AvidUMS, AvidACS, etc.
- <*node*> (optional) is the node of interest. Omitting the node cleans up the resource on all nodes.

If you receive an “object/attribute does not exist” error message, it indicates the resource is active on more than one node. Repeat the command using the group name for the resource (the “everywhere” form). For example, for the AvidAll resource, use AvidAllEverywhere. For AvidConnectivityMon, use AvidConnectivityMonEverywhere. Services contained in the postgres resource group (postgres_fs, AvidClusterIP and pgsqlDB) can be addressed individually, or as a group.

It is important to clear the failures as this also clears the fail-counts. Should a resource fail enough times on the master node to reach the migration-threshold, Pacemaker will remove the node from the cluster and failover to the slave node. If the cluster remains unsupervised, fail-counts could eventually lead to an unexpected failover and a temporary loss of client communication.

When troubleshooting, it may be necessary to stop, start or restart a resource. This can be accomplished with the following commands:

```
crm resource stop <resource-name>
crm resource start <resource name>
crm resource restart <resource-name>
```

As a reminder, press CTRL-C on a Windows keyboard or CMD-C on a Mac keyboard to exit the crm_mon console.

Interpreting Failures in the Cluster

The following section provide additional details on what users should expect from service, resource or node failures.

What impact does a failover have upon users?

Most service failures result in an immediate service restart on the same node in the cluster. In such cases, users generally do not notice the failure. At worst, their attempts to interact with the service in question may return errors for a few seconds but full functionality is quickly restored with no data loss.

If a service fails enough times to reach the failure threshold, the node is removed from the cluster. During this 20-30 second period, users will experience errors until the new master node takes over. If a user loses patience and leaves the page or closes the browser they may lose unsaved changes.

Do I need to investigate every time I see a fail-count?

No. Most service failures are due to temporary software issues. Services are quickly restarted by the cluster and users may not ever experience an interruption of service. If the fail-count appears to be the result of a benign service failure, simply reset the service's failure count. Monitoring the fail-counts ensures that future failures will not trigger a failover. If a service or resource continually fails, the issue should be investigated further.

How important are failovers?

In most cases service failures are benign, and the automated restart is sufficient. You may want to monitor cluster status regularly. If services on some nodes are occasionally reporting a fail-count of 1, take some initiative to verify that server hardware is OK, and that disk space is not compromised. You can even look at the time of the failure and retrieve logs.

However, a node may have failed because of a lack of disk space or a hardware failure, in which cases it should only be added back to the cluster only after it has been repaired.

7 System Administration

This chapter contains administrative tasks for an MediaCentral Platform Services that are not directly related to a new installation or upgrade. This chapter is divided into two sections: one for Single Server Administration and another for Cluster Administration.

Single Server Administration

This section relates includes processes that are specific to single-server installations and do not apply to cluster configurations.

Shutting Down or Rebooting a Single Server

Unlike a cluster, there are no prerequisite steps or concerns when shutting down, rebooting, or starting up a single server.

To shutdown the server, enter the following command:

```
shutdown -h now
```

To simply reboot a single server, enter the following command:

```
reboot
```

Cluster Administration

This section relates includes processes that are specific to cluster installations and do not apply to single-server configurations. The following processes are included:

- Reviewing the Cluster Configuration File
- Temporarily Removing a Node
- Permanently Removing a Node
- Adding Nodes to a Cluster
- Changing the Administrator E-mail Address
- Changing IP Addresses in a Cluster
- Taking Nodes Offline and Forcing a Failover
- Shutting Down or Rebooting a Single Cluster Node
- Shutting Down the Cluster
- Starting the Cluster
- Performing a Rolling Reboot

Reviewing the Cluster Configuration File

During the cluster installation, a configuration file was created which contains information about the cluster and the resources managed by Pacemaker. You can review the contents of the configuration file at any time by typing:

```
crm configure show
```

For example, the AvidClusterIP primitive contains the cluster IP address and the network interface being used (e.g. eth0).

If necessary, press Q to get back to the Linux command line prompt.

The name and location of the cluster configuration file is: `/etc/crm/crm.conf`

However, when running the "show" command, the output sent to the screen is actually contained in the Pacemaker configuration file:

`/var/lib/pacemaker/cib/cib.xml`

Temporarily Removing a Node

New installations, system upgrades or troubleshooting might require you to take actions on the node that would affect the cluster. In these cases, the node should be temporarily removed from the cluster to avoid introducing service failures. This can be accomplished by either removing the node from the cluster or stopping the clustering services.

Taking the master node offline using either of the following two processes will initiate a failover.

To Temporarily Remove a Cluster Node:

- A node can be temporarily removed from the cluster using the cluster resource manager:

  ```
  crm node standby <node>
  ```

  Putting a node into standby informs pacemaker that you want to temporarily remove this node's ability to host resources. The Pacemaker and Corosync services are still running on a standby node which is why you can still interact with it through `crm` commands.

- Alternatively, stopping Pacemaker and Corosync will also take the node offline:

  ```
  service pacemaker stop && service corosync stop
  ```

To Restart the Clustering Services:

- If you issued the "node standby" command, bring the node back online with the following command:

  ```
  crm node online <node>
  ```

  Open the `crm_mon` utility on another node and watch as the node comes online and starts the appropriate resources.

- If you stopped the pacemaker and corosync services, use the following command to restart the services and bring the node back online:

  ```
  service corosync start && service pacemaker start
  ```

  Notice that this command restarts the two services in the reverse order in which they were stopped.

Permanently Removing a Node

As discussed, a node can be temporarily removed from the cluster by putting it into standby. This section outlines the process of removing a node that will not rejoin the cluster configuration. The following is an overview of the steps required to remove a node:

- Removing the Node from Sharded Mongo
- Removing the Node from the Corosync Cluster
- Removing the Node from GlusterFS

The following process applies to the removal of a load-balancing node. If you need to remove the slave node from the cluster, Avid recommends backing-up all system settings, re-imaging the nodes, and re-creating the cluster.

Removing the Node from Sharded Mongo

If the node serves as a sharded Mongo arbiter, the sharded Mongo configuration must be updated. To verify if the node is a sharded Mongo arbiter, see “Obtaining the Status of Sharded Mongo” in the *MediaCentral Platform Services Installation and Configuration Guide* v2.8 or later.

For complete details on removing the sharded Mongo arbiter, see “Uninstalling the Sharded Mongo Arbiter” in the *MediaCentral Platform Services Installation and Configuration Guide* v2.8 or later.

Removing the Node from the Corosync Cluster

Permanently removing a node involves a reconfiguration of the Corosync / Pacemaker cluster as well as removal of the node from the RabbitMQ cluster.

To Remove a Node from the Corosync Cluster

1. The Corosync cluster should appear healthy (no failures / all resources available) prior to beginning this process. Open the Cluster Resource Monitor to verify the cluster status:

   ```
   crm_mon -f
   ```

   Press CTRL-C on the keyboard to exit the Cluster Resource Monitor.

2. Bring the cluster into maintenance mode by putting each node into standby with the following command:

   ```
   crm node standby <node name>
   ```

   Start with the load-balancing nodes, then the slave node and finally the master node.

3. Stop the cluster services on the node you need to remove:

   ```
   service pacemaker stop
   service corosync stop
   ```

4. From any cluster node other than the one you are removing, delete the node that you want to remove:

   ```
   crm node delete <node name>
   ```

   The system will respond with the following:

   ```
   INFO: node <node name> deleted
   ```

5. Prior to bringing the Corosync cluster back online, the node must also be removed from the RabbitMQ cluster.

   a. Check the current status of the rabbitmq cluster:

   ```
   rabbitmqctl cluster_status
   ```

   All cluster nodes, including the one you want to remove should be listed. Example:

   ```
   [root@wavd-mcs02 etc]# rabbitmqctl cluster_status
   Cluster status of node 'rabbit@wavd-mcs02' ...
   [{nodes,[{disc,['rabbit@wavd-mcs01','rabbit@wavd-mcs02',
                   'rabbit@wavd-mcs03']}]},
    {running_nodes,['rabbit@wavd-mcs01','rabbit@wavd-mcs02']},
    {cluster_name,<<"rabbit@wavd-mcs01">>},
    {partitions,[]}]
   ...done.
   ```

   b. Stop the rabbitmq service on the node to be removed:

   ```
   service rabbitmq-server stop
   ```

   c. From any cluster node other than the one you are removing, remove the node from rabbitmq:

   ```
   rabbitmqctl forget_cluster_node rabbit@<node name>
   ```

   d. Check the status of the rabbitmq cluster again:

   ```
   rabbitmqctl cluster_status
   ```

   Rabbitmq should no longer list the removed node. Example:

   ```
   [root@wavd-mcs02 etc]# rabbitmqctl cluster_status
   Cluster status of node 'rabbit@wavd-mcs02' ...
   [{nodes,[{disc,['rabbit@wavd-mcs01','rabbit@wavd-mcs02']}]},
    {running_nodes,['rabbit@wavd-mcs01','rabbit@wavd-mcs02']},
    {cluster_name,<<"rabbit@wavd-mcs01">>},
    {partitions,[]}]
   ...done.
   ```

6. Update the hosts file on all nodes to eliminate the deleted node.

   For details, see “Verifying the hosts File Contents” in the *MediaCentral Platform Services Installation and Configuration Guide*.

You may also want to take this opportunity to remove the deleted node information from your site’s DNS server.

7. Run the `cluster setup-cluster` command on the master node. This is required to update the cluster with the list of nodes to be excluded from DRBD.

   For detailed instructions on using this command, see “Starting the Cluster Services on the Master Node” in the *MediaCentral Platform Services Installation and Configuration Guide*.

   This command will bring the cluster back online.

8. Open the Cluster Resource Monitor to verify the status of the cluster:

   ```
   crm_mon -f
   ```

The number of “Nodes configured” and the number of “expected votes” should match the number of actual nodes in your cluster (one less than before).

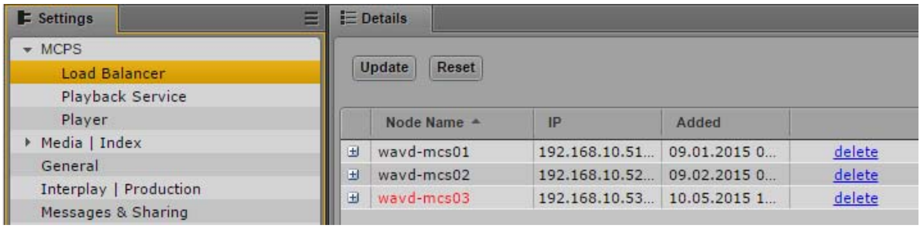
9. The node is now removed from the cluster. However, a residual reference to the node might still exist in the “Load Balancer” section of MediaCentral UX. If this reference exists, it should be removed.

   a. Sign in to MediaCentral UX as a user with administrative privileges.

   b. Select “System Settings” from the Layout selector.

   c. Select “Load Balancer” under MCPS from the left side of the interface.

   d. Click the delete button next to the node you have removed from the cluster. In the example below, “wavd-mcs03” has been removed:



	Node Name	IP	Added	
⊞	wavd-mcs01	192.168.10.51...	09.01.2015 0...	delete
⊞	wavd-mcs02	192.168.10.52...	09.02.2015 0...	delete
⊞	wavd-mcs03	192.168.10.53...	10.05.2015 1...	delete

   e. You will be asked to confirm you want to delete the node.

   Click the Yes button.

After performing the above steps, a “node offline” message may reappear in the cluster monitoring tool (crm_mon) after the first reboot of the cluster following the removal process. To eliminate the “ghost” node, delete node from the cluster by repeating the `crm node delete <node>` *command.*

Removing the Node from GlusterFS

If the GlusterFS file replication service is in use, the node must be removed from the Gluster configuration.

To Remove a Node from GlusterFS

1. Unmount the Gluster volumes on the node that you want to remove:

```
umount /cache/download
umount /cache/fl_cache
umount /cache/render
```

2. Similar to the `gluster volume create` command used in the “Configuring the GlusterFS Volumes” process found in version 2.4 of the *MediaCentral Platform Services Installation and Configuration Guide* you will use the `remove-brick` command to remove the node from Gluster. Complete this step on a node other than the one you are removing:

```
gluster volume remove-brick gl-cache-dl replica N hostname:/cache/gluster/
gluster_data_download force
```

```
gluster volume remove-brick gl-cache-fl replica N hostname:/cache/gluster/
gluster_data_fl_cache force
```

```
gluster volume remove-brick gl-cache-mcam replica N hostname:/cache/
gluster/gluster_data_multicam force
```

In the above command:

- "N" is the total number of nodes (minus the node you are removing). If you have 4 nodes and you are removing one, the command would include: `replica 3`.
- "hostname" is the short host name of the cluster node you want to remove.

After each of these commands, you will receive the following message:

```
Removing brick(s) can result in data loss. Do you want to Continue? (y/n)
```

Enter "y" (without the quotes) to confirm. Through this command, you are telling Gluster that you want one less copy of the replicated data. Gluster wants you to confirm that you understand that the data will be lost on the removed node.

You can monitor the progress of the removal process with the following command:

```
watch gluster volume remove-brick <volume> replica N hostname:/<share>
```

3. Once the remove-brick process is complete for all three volumes, verify the number of Gluster peers.

   ```
   gluster peer status
   ```

   At this time, all Gluster peers should still be listed.

4. Remove the node from Gluster with the following command:

   ```
   gluster peer detach <node name>
   ```

5. Repeat the `gluster peer status` command and verify the removed node is no longer present.

The removed node will contain many lingering components of the MCS installation including manually edited system files, network information and more. Depending on what you intend to do with the removed node, you may want to consider re-imaging the server to avoid any conflicts in the event that it is placed back into production.

Adding Nodes to a Cluster

Additional nodes are often added to existing MCS clusters to add horizontal scale which accommodates increased client capacity and system load. Although the process to add a node to an existing cluster is similar to that of an entirely fresh installation, certain steps might be different as the new installation process has changed over the life of the product.

Review and complete the following sections to add a node to an existing cluster:

- Adding Nodes to the Corosync Cluster
- Adding Nodes to GlusterFS
- Configuring Linux User Accounts
- Reconfiguring Sharded MongoDB
- Additional Procedures

Adding Nodes to the Corosync Cluster

To add node(s) to the Corosync cluster:

1. Build the new node by completing the following sections of the MCS Install Guide:
   - Installation Prerequisites
   - BIOS and RAID Configuration
   - Software Installation

When updating the /etc/hosts file, remember to add the new node to the hosts file of all other cluster nodes.

2. Review the “Cluster Overview” section of the “Clustering” chapter in the MCS Install Guide and verify that all prerequisites have been met.
3. From the master node only, run the `cluster setup-cluster` script to specify the existing and new non-drbd node(s):

   ```
   /opt/avid/cluster/bin/cluster setup-cluster --cluster_ip="cluster IP
   address" --pingable_ip="router IP address" --cluster_ip_iface="eth0" --
   admin_email="comma separated e-mail list" --drbd_exclude="comma separated
   list of non-DRBD nodes"
   ```

   Review the MCS Install Guide for details on the exact usage of this command. The syntax of the command is very important.
4. Open the Cluster Resource Monitor on the master node:

   ```
   crm_mon -f
   ```

   The next step in this section adds the new node to the cluster. The Cluster Resource Monitor enables you to watch the progress of that process.

5. On the new node, complete one of the following commands:

If your existing cluster has been configured for unicast communication, you must update the Corosync configuration on this new node. Complete step 5a below and then refer to step 2 and later of the "Unicast Support in Clustering" process in the Avid MediaCentral Platform Services Installation and Configuration Guide.

   a. If your network has no other multicast activity, the default multicast address of 239.192.1.1 is assumed in the following command:

```
/opt/avid/cluster/bin/cluster setup-corosync --corosync-bind-iface=interface --rabbitmq_master="master hostname"
```

   b. If your network includes multicast activity (perhaps a second MCS system), specify a custom multicast address with the following command:

```
/opt/avid/cluster/bin/cluster setup-corosync --corosync-bind-iface=interface --corosync-mcast-addr="multicast address" --rabbitmq_master="master hostname"
```

   Review the MCS Install Guide for details on the exact usage of this command. The syntax of the command is very important.

   As before, messages appear echoing the Corosync network binding process. Various services are temporarily shut down and restarted. A message appears indicating the Corosync cluster engine has successfully started.

   The following is sample output:

```
bindip=192.168.10.53/24 bind_iface=eth0 bind_network=192.168.10.0
mcast_addr=239.192.1.1

.....
Starting Corosync Cluster Engine (corosync):                   [  OK  ]
Starting Pacemaker Cluster Manager                             [  OK  ]
```

6. Starting with the master node, restart the following services so they register correctly on the newly created instance of the message bus:

```
service avid-acs-messenger restart
```

```
service avid-aaf-gen restart
```

   If running MCS v2.7 or later, you must also restart the avid-acs-mail service:

```
service avid-acs-mail restart
```

   Repeat this step on all cluster nodes to restart the services on each node.

Adding Nodes to GlusterFS

If GlusterFS volume replication has been configured on the existing nodes, Gluster needs to be configured on the new node(s) as well. The process to configure GlusterFS changed in MCS v2.5. This section includes processes for reconfiguring Gluster for systems running v2.5 and later as well as v2.4 and earlier; be sure to follow the correct process for your version of MCS.

To reconfigure GlusterFS in MCS v2.5 and later:

1. On the new node, verify that the Gluster daemon, glusterd, is running:

```
service glusterd status
```

If the service is not running, start it manually:

```
service glusterd start
```

2. On the new node, create the RHEL physical directories that Gluster will use to build its GlusterFS file system:

   ```
   mkdir -p /cache/gluster/gluster_data_download
   mkdir -p /cache/gluster/gluster_data_fl_cache
   mkdir -p /cache/gluster/gluster_data_multicam
   ```

3. From the Corosync master node, run the Gluster configuration script:

   ```
   /opt/avid/cluster/bin/gluster_setup
   ```

   The process adds the new node to the configuration. Once the script is complete, you should receive a confirmation message:

   ```
   Starting glusterd:                                         [  OK  ]
   INSTALLATION OF GLUSTERFS FINISHED
   ```

4. Run the same Gluster configuration script on the new node. The process should complete with the same confirmation message as reported on the master node.

5. Verify that you the nodes are aware of each other with the following command:

   ```
   gluster peer status
   ```

   The system responds by indicating the number of peers, their host names and connection status, plus other information. Example:

   ```
   Number of Peers: 2

   Hostname: wavd-mcs02
   Uuid: 220976c3-dc58-4cdc-bda3-7b2213d659fc
   State: Peer in Cluster (Connected)

   Hostname: wavd-mcs01.wavd.com
   Uuid: 21ab6e19-2d41-4333-bc39-3ace3bf16c77
   State: Peer in Cluster (Connected)
   ```

6. On the new node, test file replication by creating two test files in the `/cache` directories:

   ```
   touch /cache/download/hello01.txt
   touch /cache/render/hello02.txt
   ```

7. On all other nodes, verify that the files have been replicated to the local `/cache` directories:

   ```
   ls /cache/download/
   ls /cache/render/
   ```

To reconfigure GlusterFS in MCS v2.4 and earlier:

1. Complete “Starting GlusterFS” in the MCS Install Guide.

   In this process, “MCS Install Guide” refers to the v2.4 *MediaCentral Platform Services Installation and Configuration Guide*.

2. Complete “Creating the Trusted Storage Pool” in the MCS Install Guide. Only the new node or nodes need to be probed.

3. Similar to the `gluster volume create` command used in the “Configuring the GlusterFS Volumes” process found in the MCS Install Guide you will use the `add-brick` command to add the new node to Gluster. Complete this step on a node other than the one you are adding.

```
gluster volume add-brick gl-cache-dl replica N hostname:/cache/gluster/
gluster_data_download

gluster volume add-brick gl-cache-fl replica N hostname:/cache/gluster/
gluster_data_fl_cache

gluster volume add-brick gl-cache-mcam replica N hostname:/cache/gluster/
gluster_data_multicam
```

In the above command:

- "N" is the total number of nodes (including the new node).
- "hostname" is the short host name of the new cluster node.

If needed, this command can be used to add multiple nodes to Gluster at the same time by specifying additional host names.

4. Complete the following sections in the MCS Install Guide for configuring Gluster:
   - "Setting Gluster Volume Ownership"
   - "Making the RHEL Cache Directories"
   - "Changing Ownership and Mounting the GlusterFS Volumes"
   - "Testing the Cache"
   - "Ensuring Gluster is On at Boot"

If the Gluster Add Node process fails, see "Deleting the Gluster Volume and Bricks" on the Avid Knowledge Base article for troubleshooting assistance.

Configuring Linux User Accounts

When adding a new node to an existing cluster, the UID (user ID) and GID (group ID) of the mongod, redis, and maxmin users must be the same across all nodes. If the IDs do not match between the nodes, DRBD replication failures, Gluster replication failures, or playback issues could arise. To begin to address this issue, new installations of MediaCentral Platform Services v2.7 set the UID and GID of the redis user to a fixed value of 148. MCS v2.8 sets the ID for the mongod user to a fixed value of 184, and MCS v2.9 sets the maxmin user to a fixed ID of 205.

The fixed IDs are automatically configured for new installs only. If upgrading the MCS software from an earlier version, the IDs on existing nodes are not altered.

Since MCS v2.7 and later sets fixed values for some users, customers with existing clusters that are adding a new node might have different values for the users across the cluster nodes. To address this, Avid has created a script that checks and sets the UID and GID of all three user accounts.

This script is included in MCS v2.7 and later, however only the script included in v2.8.1 and later analyzes the ID of all three users. If necessary, the script from v2.8.1 and later can be used with prior versions of MediaCentral Platform Services.

The script is located at: `/opt/avid/installer/conf/avid_fix_UID.sh`

To verify the user IDs:

1. Install and configure the new cluster node per the instructions in the *MediaCentral Platform Services Installation and Configuration Guide*.

2. From the cluster master node, use the “id” command to check the UID and GID of each of the three users:

   ```
   id mongod
   id redis
   id maxmin
   ```

   The system reports the UID and GID for each user as shown in the following example:

   ```
   [root@wavd-mcs01 ~]# id mongod
   uid=184(mongod) gid=184(mongod) groups=184(mongod)
   ```

3. Repeat the “id” command for each user (mongod, redis, maxmin) on every cluster node. Verify that the UID and GID for each of the three users is the same on all nodes.

   If the IDs are already the same on each node, no further action is required. If the ID’s are different between the nodes, continue to the next step to adjust the values.

4. From the cluster master node, run the avid_fix_UID script with the appropriate option to fix the user ID:

   ```
   avid_fix_UID -[options]
   ```

   - The **-m** option sets the mongod user to a fixed value of 184.
   - The **-r** option sets the redis user to a fixed value of 148.
   - The **-x** option sets the maxmin user to a fixed value of 205.
   - The **-h** option shows the help page for this script.

   If multiple users need to be adjusted, you can specify multiple options at once. For example the following command simultaneously alters the ID for both the mongod and redis users:

   ```
   avid_fix_UID -mr
   ```

   Running the script with no options simply reports the current status of all three users as shown in the following example:

   ```
   [root@wavd-mcs01 ~]# avid_fix_UID
   SUCCEEDED!!!    The UID mongod is 184
   SUCCEEDED!!!    The GID mongod is 184
   FAILED!!!       The UID redis  is 492 and not 148. Use option -r for fix
   FAILED!!!       The GID redis  is 492 and not 148
   FAILED!!!       The UID maxmin is 206 and not 205. Use option -x for fix
   FAILED!!!       The GID maxmin is 206 and not 205
   ```

   If the users are assigned ID’s other than the assigned fixed values, they are reported as “FAILED”. However, a FAILED message does not necessarily indicate there is a problem. As long as the IDs for each user match across each cluster node, the ID can be any value. For example, if you have a three node cluster and the UID of the redis user is set to 96 on all three nodes, the configuration is acceptable.

   When the script is executed, it first verifies that the IDs to be used are not already assigned to another user. If the ID is in use, the script reports the following message:

   ```
   “The UID <ID> is already in use and cannot be used for the user <user>.
   Please modify the variable at the beginning of the avid_fix_UID script
   (located in /opt/avid/bin/) and configure an unused UID value.”
   ```

   If necessary, use the Linux “vi” editor to change the default IDs in the script to alternate, unused values. To see a list of all users and their respective UID/GID, enter: `cat /etc/passwd`

5. Run the avid_fix_UID script on each node, specifying the correct options to fix the ID's for the required users.

If fixing the maxmin user, the script stops the AvidICPS resource and underlying avid-icps-manager service. It also changes the ownership of the `/cache/fl_cache` *and* `/cache/download` *directories. If the cache directories contain many files, be patient as this process can take time.*

6. Finally, repeat the "id" command on each node to verify that the UID and GID of each of the three users matches across the cluster:

   ```
   id mongod
   id redis
   id maxmin
   ```

7. For cluster configurations with GlusterFS, run the Gluster setup script to update the Gluster configuration with the new UID information. Complete this step once from any cluster node:

   ```
   /opt/avid/cluster/bin/gluster_setup
   ```

Reconfiguring Sharded MongoDB

If you are adding nodes to MCS v2.6 or later, you must reconfigure the sharded Mongo environment in the following scenarios:

- Adding a third node to a two-node, non-multi-zone cluster running MCS V2.6 or later. The original two-node cluster would have an external arbiter configured. When adding a third node, it is preferable to have the load-balancing node serve as the sharded Mongo arbiter.
- Any system running MCS v2.9 or later. The sharded Mongo ansible hosts file in MCS v2.9 includes information on all nodes. The configuration must be recreated to include the new node.

At a high-level, this process involves the following steps:

- (if applicable) Remove the non-Avid Linux or Windows arbiter from the configuration.
- Clean the sharded Mongo configuration on the new node (ONLY on the new node):

  ```
  mongo-clean-local
  ```

- Run the configuration script on the sharded Mongo management node:

  ```
  mongo-create-configuration -c
  ```

- Mount the RHEL ISO on the sharded Mongo management node.
- From the sharded Mongo management node, run the final playbook script to reconfigure the sharded Mongo environment:

  ```
  mongo-playbook-setup
  ```

For detailed instructions on removing or adding nodes to the sharded MongoDB environment, see the *MediaCentral Platform Services Installation and Configuration Guide*.

Additional Procedures

This section includes additional procedures that might or might not apply to your configuration. Review the following items and complete processes that apply to your system:

- If you are running MCS v2.7 or later and you have altered avid-acs-gateway configuration file to allow external systems to access MCS, update the configuration file on the new node. For more information, see “Configuring Access for External Systems” in the *MediaCentral Platform Services Installation and Configuration Guide*.
- If you have Asset Watermarking enabled with MCS v2.7 or later, refer to “Enabling Asset Watermarking” in the *MediaCentral Platform Services Installation and Configuration Guide* to enable this option on the new node.
- If your cluster has Media Index enabled, the configuration must be updated to include the new node. For more information, refer to the *Avid Media | Index Configuration Guide*.

Changing the Administrator E-mail Address

When you set up the cluster, you provided an administrator e-mail address where the system sends e-mails related to cluster performance. You can change the e-mail address (or add others) at any time using the Corosync-Pacemaker command-line interface for configuration and management, *crm*.

Be careful when editing the cluster configuration settings. Incorrect settings will break the cluster.

To change the cluster administrator e-mail address:

1. The e-mail address information is stored in the crm configuration file. Edit the file with the following command:

   ```
   crm configure edit
   ```

Due to a bug in the Cluster Resource Manager, "crm configure edit" must be entered on one line. Do not enter the Cluster Resource Manager in steps (that is crm -> configure -> edit). If you do, the changes are not saved.

2. Scroll to the end of the file or press "Shift-g" to jump to the end of the file.
3. Find the line containing the cluster administrator e-mail address. Example:

   ```
   rsc_defaults rsc_defaults-options: \
           admin-email="admin@wavd.com"
   ```

4. Alter the existing e-mail address or add additional e-mail addresses by separating each contact with a comma. Example:

   ```
   rsc_defaults rsc_defaults-options: \
           admin-email="admin@wavd.com,engineering@wavd.com"
   ```

5. Save the changes using the same command as you would use in a "vi" edit session.

   Press <ESC> and type: :**wq**

   Alternatively, if you do not want to save your changes, press <ESC> and type **:q!**

6. The system responds by writing the updated configuration file to a temporary location and outputting an error message similar to the following:

   ```
   "/tmp/tmpjve4D9" 72L, 3258C written
   ERROR: rsc-options: attribute admin-email does not exist
   Do you still want to commit?
   ```

7. Type **yes** (the entire word) to commit your changes.
8. Verify the changes have been made by displaying the Cluster Resource Manager configuration information:

   ```
   crm configure show
   ```

   Press **q** to exit.

9. The new e-mail address(es) are now active.

Changing IP Addresses in a Cluster

In the event that you need to alter the IP address of a node or an entire cluster, follow the procedures below as they apply to your network change requirements.

Recall that a cluster has multiple IP addresses:

- Node IP addresses. Each node is a assigned a standard unicast address.
- Cluster IP address. This address is used by the nodes to communicate with each other within the cluster. By default, this is a multicast address. However, additional steps can be taken to alter the configuration with a unicast address.
- Virtual IP address. This is a unicast address that systems outside of the cluster use to identify the MCS system.

Once all changes have been made, remember to update any external systems that may have used an IP address to locate MediaCentral Platform Services. Examples include:

- MediaCentral UX clients and Media Composer Cloud clients
- IP address information contained in SSL certificates used with web browsers
- Configuration file for the MediaCentral UX Desktop application
- Interplay Administrator settings
- Settings configured during a Media Distribute installation

Also remember to update any DNS servers which contain forward and reverse entries for the MCS systems.

The procedures below may disconnect your server from the network. It may be necessary to complete these steps from a direct KVM connection to the MCS servers.

Changing the IP Addresses within a Cluster

1. Stop the cluster services on all nodes. Start with the load-balancing nodes, then the slave node and finally the master node:

   ```
   service pacemaker stop
   service corosync stop
   ```

2. Proceed to one or more of the following sections:

   - If you need to alter the node IP address(es), see Changing the Node IP Address(s) below.
   - If you need to alter the multicast address assigned to the cluster, see Changing the Cluster IP Address below.
   - If you need to alter the virtual cluster IP address, see Changing the Virtual IP Address below.

   Once all required changes have been made, continue with step 3 of this process.

3. Bring the cluster back online on the master node:

   ```
   service pacemaker start
   service corosync start
   ```

4. Open the Cluster Resource Monitor to verify the status of the cluster:

   ```
   crm_mon -f
   ```

   Wait for the master node to start all resources.

5. Bring the slave and load-balancing nodes back online:

   ```
   service pacemaker start
   service corosync start
   ```

   Watch the Cluster Resource Monitor to ensure that all resources start normally.

6. If your changes are complete, verify your changes by testing basic functionality of the MCS system.

Changing the Node IP Address(s)

1. Review and update the contents of the hosts file.

   See "Verifying the hosts File Contents" in the *MediaCentral Platform Services Installation and Configuration Guide* for instructions on altering the hosts file.

2. Update the network interface configuration file:

   ```
   vi /etc/sysconfig/network-scripts/ifcfg-eth0
   ```

In the example above "eth0" represents the primary network adapter. On a Dell server, "eth0" would be replaced with "em1", p1p1", or "p2p1".

3. Edit the lines containing the site-specific network information. Example:

   ```
   IPADDR=192.168.10.51

   NETMASK=255.255.255.0

   DNS2=192.168.10.20

   GATEWAY=192.168.10.1

   DNS1=192.168.10.10
   ```

4. Save and exit the vi session. Press <ESC> and type: **:wq**

5. Restart the network service:

   ```
   service network restart
   ```

6. If you are changing the IP address of the master and / or slave nodes, you must edit the drbd configuration file.

   a. Open the file for editing:

   ```
   vi /etc/drbd.d/r0.res
   ```

   b. Find and change the IP address(es) associated with the altered node(s):

   ```
   on wavd-mcs02 {
       device    /dev/drbd1;
       disk      /dev/sda2;
       address   192.168.10.52:7789;
       meta-disk internal;
     }
   on wavd-mcs01 {
       device    /dev/drbd1;
       disk      /dev/sda2;
       address   192.168.10.51:7789;
       meta-disk internal;
     }
   }
   ```

   c. Save and exit the vi session. Press <ESC> and type: **:wq**

7. Return to step 2 of the "Changing the IP Addresses within a Cluster" process.

Changing the Cluster IP Address

1. Update the Corosync configuration file to include your updated IP information:

   ```
   vi /etc/corosync/corosync.conf
   ```

   Important fields include:

   - bindnetaddr ("pingable_ip" address used in multicast and unicast configurations)
   - mcastaddr (multicast IP used in multicast configurations)
   - memberaddr (unicast IP addresses used in unicast configurations)

   See "Unicast Support in Clustering" the *MediaCentral Platform Services Installation and Configuration Guide* for an example of a `corosync.conf` file configured for multicast and unicast.

2. Save and exit the vi session. Press <ESC> and type: **`:wq`**

3. Return to step 2 of the "Changing the IP Addresses within a Cluster" process.

Changing the Virtual IP Address

1. On the Master node, run the `cluster setup-cluster` command with your updated IP address information to update the cluster configuration file.

   See "Starting the Cluster Services on the Master Node" the *MediaCentral Platform Services Installation and Configuration Guide* for details.

   This command will start the cluster services on the master node.

2. If the virtual IP address has been added to the hosts file, review and update the contents of the hosts file on all nodes.

   See "Verifying the hosts File Contents" in the *MediaCentral Platform Services Installation and Configuration Guide* for instructions on altering the hosts file.

3. Restart the following services so they register correctly on the newly created instance of the message bus:

   ```
   service avid-acs-messenger restart
   ```

   ```
   service avid-aaf-gen restart
   ```

4. Open the Cluster Resource Monitor and wait for the resources to start on the master node.

   ```
   crm_mon -f
   ```

   It may be useful to keep the *crm_mon* tool open as additional nodes join the cluster.

5. On the slave and load-balancing nodes, follow the process for "Adding Nodes to the Cluster" in the *MediaCentral Platform Services Installation and Configuration Guide*.

6. Return to step 6 of the "Changing the IP Addresses within a Cluster" process.

Taking Nodes Offline and Forcing a Failover

At times it might be required to take a node offline for troubleshooting. Pacemaker offers an easy way to temporarily remove and reactivate a node in the cluster. The same commands can be used to force a failover of the cluster which is useful when testing a fully functional system.

Be aware that since the playback service is load-balanced across all cluster nodes, taking a node offline can result in an interruption in playback. If this happens, the client will automatically be redirected to another node to service the playback request.

To temporarily remove and reactivate a cluster node:

The `standby` command can be used to temporarily remove a node from the cluster:

```
crm node standby <node name>
```

If you are watching the CRM utility, the cluster will update and the status of the node will appear near the beginning of the monitor window. As seen in the following example, the node's status will change from "online" to "standby":

```
Node wavd-mcs02: standby
Online: [ wavd-mcs01 ]
```

Since the node can still be contacted by the cluster, it does not appear as "offline".

The `online` command is used to rejoin the node to the cluster:

```
crm node online <node name>
```

This will bring the node back online. As with the standby process, the CRM utility will update the status of the node to "online" and the appropriate services will be started.

To force a failover in the cluster:

Using the "standby" command on master node of the cluster will result in a failover event. This is an effective way to verify that the cluster is working as expected. Follow the process below to force a failover to the slave node, and if desired, to reverse the process.

Forcing a failover will disconnect all MediaCentral UX clients currently logged into the system. Ensure that all users are made aware that a failover will take place and that they should save all work. Any active MediaCentral UX client sessions will be logged out and users will receive a message indicating that they need to log back in.

1. Log in to any node in the cluster as *root* and open the Cluster Resource Monitor utility:

   ```
   crm_mon -f
   ```

   This returns the status of all cluster-related services on all nodes. Ensure all nodes are active and operating normally prior to the test. Any failures should be cleared or investigated and cleared so as not to initiate additional unexpected failovers.

2. Note the line identifying the master node:

   ```
    AvidClusterIP      (ocf::heartbeat:IPaddr2):    Started wavd-mcs01
   ```

3. In a separate terminal session log in to any non-master node as *root* and put the master node into standby mode:

   ```
   crm node standby <hostname>
   ```

   In the above command, replace `<hostname>` with the host name of the master node (e.g. *wavd-mcs01*).

4. Observe the failover in the *crm_mon* utility as the former slave node is reassigned as the new master.

5. Once all resources have started on the new master node, bring the standby node back online:

   ```
   crm node online <original master hostname>
   ```

6. Observe in the *crm_mon* window as the node is brought back up and rejoins the cluster as the slave node.

When the master node (e.g. node-1) is taken offline and brought back online again in Interplay Central v1.x, an additional failover occurs and the original master node (e.g. node-1) becomes the master again. This behavior changed in MCS 2.x to allow the new master node (e.g. node-2) to remain the new master node. This behavior is reliable in two node clusters, but failover back to the master node could still occur in clusters with three or more nodes.

7. If you want to restore the original node to the role of master, temporarily put the current master into standby mode, so control fails over again, back to the original master node.

Shutting Down or Rebooting a Single Cluster Node

The Linux reboot process is thorough and robust, and automatically shuts down and restarts all the MCS and clustering infrastructure services on a server in the correct order. However, when the server is a node in an MCS cluster, care must be taken to remove the node from the cluster — that is, stop all clustering activity first — before shutting down or rebooting the individual node.

Failing to observe the correct procedures can have unexpected consequences including unexpected failover events, loss of node connectivity to the cluster or complete loss of client connectivity to MCS.

Before taking any cluster nodes offline, alert users of the event. If applicable, users should save all work prior to the shutdown or reboot procedure.

Verify the RabbitMQ cluster

1. Verify if the RabbitMQ cluster is active and lists all nodes:

   ```
   rabbitmqctl cluster_status
   ```

   The following is an example of how a 2-node cluster should appear. The two nodes names appear on both the "`nodes`" and "`running_nodes`" lines.

   ```
   Cluster status of node 'rabbit@wavd-mcs01' ...
   [{nodes,[{disc,['rabbit@wavd-mcs01','rabbit@wavd-mcs02']}]},
   {running_nodes,['rabbit@wavd-mcs02','rabbit@wavd-mcs01']},
   {partitions,[]}]
   ...done.
   ```

2. A normal response from the previous command is a good indicator that the RabbitMQ cluster is healthy, but to verify the status, a second command is required:

   ```
   acs-query
   ```

   - A normal output should display a long list of configuration parameters. If you see this, continue with the shutdown process.
   - If instead you receive a "request timeout", "bus is not running", "node is down" or equivalent error, it indicates that the RabbitMQ cluster is problematic. If an error occurs, see the RabbitMQ cluster troubleshooting article on the Avid Knowledge Base for guidance:

     http://avid.force.com/pkb/articles/en_US/troubleshooting/RabbitMQ-cluster-troubleshooting

Shut down or reboot the cluster node:

1. Log into the node as the Linux *root* user.
2. Stop the Pacemaker and Corosync services:

   ```
   service pacemaker stop && service corosync stop
   ```

   The services should stop with a green [OK] status.

You can safely stop these cluster services without putting the nodes in Standby. If you are stopping pacemaker and Corosync on the master node, the cluster will fail over to the slave node and it will become the cluster master. That is expected and normal behavior.

3. Once the pacemaker and Corosync services have stopped, stop the RabbitMQ service:

   ```
   service rabbitmq-server stop
   ```

   Again, the service should stop normally with a green [OK] status.

4. Once the RabbitMQ service has stopped, you can proceed with the node reboot or shutdown.

   - To reboot the cluster node:

     ```
     reboot
     ```

   - To shut down the cluster node:

     ```
     shutdown -h now
     ```

   When you power the node back up, it will automatically start the appropriate services and join the cluster. After a reboot, use the tools described in this document to inspect the Corosync cluster and the RabbitMQ cluster to confirm that all services have started normally.

   If multiple servers require a reboot, proceed one server at a time to avoid problems when the node rejoins the cluster. Once you have verified that the rebooted node is back online, reboot the next server. Continue until all required nodes are restarted.

If you had put the node into standby through the "crm node standby" command, and shut down or rebooted, the node would start the RabbitMQ service upon power-up, but the node would not rejoin the cluster. In that event, you would need to manually start the node with the "crm node online" command.

Shutting Down the Cluster

When shutting down an entire cluster, the nodes must be shut down and restarted in a specific order. Rebooting nodes in the incorrect order can cause DRBD to become confused about which node is master, resulting in a "split brain" condition. Rebooting in the incorrect order can also cause RabbitMQ to enter into a state of disarray, and hang. Both DRBD and RabbitMQ malfunctions can present misleading symptoms and can be difficult to resolve. For these reasons, a strict shutdown and reboot methodology is advised.

When shutting down and restarting an entire cluster, allow each node to power down completely before shutting down the next node.

Shutting down the cluster:

1. Use the Cluster Resource Monitor, **`crm_mon`**, to verify the current master, slave and load-balancing nodes.
2. Log into each node as the Linux *root* user.
3. Stop the pacemaker and corosync services on the load-balancing nodes. In this case, the node order is unimportant.

   ```
   service pacemaker stop && service corosync stop
   ```

4. Stop the pacemaker and corosync services on the cluster slave node.
5. Stop the pacemaker and corosync services on the cluster master node.
6. Stop the rabbitmq-server service on one load-balancing node.

   ```
   service rabbitmq-server stop
   ```

7. Shut down the server on which you just stopped the rabbitmq service

8. If you have additional load-balancing nodes, wait for the first node to be completely down, then stop the rabbitmq service and shut down the server (one at a time).
9. Once the last load-balancing node is powered-down, stop the rabbitmq service on the cluster slave node and shut down the server.
10. Once the slave node is powered-down, stop the rabbitmq service on the cluster master node and shut down the server.

Make sure you note which node was the master when you shut down the cluster. You will need this information when bringing the cluster back up.

Starting the Cluster

When bringing the cluster online, it is important to bring up the original master first. For the RabbitMQ cluster to start correctly, the last node down must be the first back up. If the nodes cannot find the “last down” node within 30 seconds of being powered-up, the RabbitMQ cluster will hang and services that depend on it — such as the ACS bus — will report errors.

To restart all cluster nodes:

1. Power-on the server that was last running as the cluster’s master node.

   Before continuing, allow this node two minutes from initial power-on to begin its boot cycle. This will ensure that key services such as RabbitMQ and the Corosync cluster start on this node first.
2. Power-on the server that was running as the cluster slave node.
3. If applicable, power-on the load-balancing nodes.
4. Once you can log into Linux on the master node, launch the Cluster Resource Monitor so that you can view progress as additional nodes join the cluster:

   ```
   crm_mon -f
   ```
5. Once all servers are up, review the Cluster Resource Monitor.
   - Confirm that the master node is running the required services.
   - Confirm all nodes are running the AvidAll and AvidICPS services.
   - If any services have failed and recovered, clear the fail-counts.

     ```
     crm resource cleanup <rsc> [<node>]
     ```
6. Using the processes outlined in “Shutting Down or Rebooting a Single Cluster Node” on page 106, verify the rabbitmq cluster is operating normally.

Performing a Rolling Reboot

A rolling reboot is a process in which one or more cluster nodes are rebooted in sequence and only one machine at a time is off-line. A rolling reboot allows the entire cluster to be restarted with minimal disruption of service to the clients.

The following list shows the correct order for a rolling reboot:

1. Power-cycle the load-balancing nodes.
2. Power-cycle the slave node.
3. Power-cycle the master node.

While a rolling reboot is minimally impactful to client operations, clients should be informed the process is taking place. Since all nodes take part in playback operations, clients will experience brief interruptions in service. When the master node is rebooted, all clients will be temporarily disconnected from MCS.

To perform a rolling shutdown / reboot:

1. From the master node, launch the Cluster Resource Monitor:

   ```
   crm_mon
   ```

2. Identify the current master and slave nodes by locating the “Master/Slave Set” information:

   ```
   Master/Slave Set: ms_drbd_postgres [drbd_postgres]
      Masters: [ wavd-mcs01 ]
      Slaves: [ wavd-mcs02 ]
   ```

3. If you have one or more load-balancing nodes, reboot one of the load-balancing nodes using the processes located in “Shutting Down or Rebooting a Single Cluster Node” on page 106.
4. Watch the CRM utility on the master node and wait for the node to rejoin the cluster and start the appropriate services.
5. If you have additional load-balancing nodes, reboot each node one at a time, allowing each node to rejoin the cluster and start services before moving on to the next node.
6. Once all load-balancing nodes have been rebooted, reboot the slave node.
7. Wait for the slave node to rejoin the cluster and start all services.
8. Close the CRM utility on the master node and open it on the slave node.
9. Reboot the master node. Watch the CRM utility as a failover to the slave node takes place.
10. Watch the CRM utility and wait for the former master node to rejoin the cluster and start all services. If any resource failures have occurred, clear them.

8 Best Practices, Troubleshooting and System Logs

This chapter provides information on maintaining your MediaCentral servers to maximize system stability and up-time. Troubleshooting procedures as well as the location and description of the log files produced by MediaCentral are also detailed in this section.

Best Practices for MediaCentral

MediaCentral Platform Services is built on Linux with server-class hardware and multiple redundant systems, making it a very robust product. Early common practices when working with MCS was to configure the system and generally leave it alone unless there was a specific issue that required attention. While this statement is mostly true, a certain level of system monitoring can further increase the amount of system up-time.

The following sections detail the daily, weekly and monthly maintenance procedures recommended by Avid. The completion frequency for these steps can be adjusted as deemed appropriate for your installation. If for example, you find that clock drift is rare in your environment, intervals between time synchronization checks could be increased.

While Avid does not recommend restarting the MediaCentral servers as part of a routine maintenance schedule, other systems such as Interplay Production servers or Avid shared storage systems are often included. If you are cycling power or restarting systems that integrate with MediaCentral Platform Services, verify connection to these systems through MediaCentral UX once the restart is complete. If users encounter any issues connecting to the back-end systems, review the following:

- If you cannot connect to the Interplay Production database after restarting the Interplay Production servers, enter one of the following commands on the MCS server:
  - On a single server: **`service avid-interplay-central restart`**
  - On a cluster, from any node: **`crm resource restart AvidIPC`**
- If you cannot connect to Avid shared storage and/or the Media pane displays a Media Offline message, enter one of the following commands on the MCS server:
  - On a single server: **`service avid-all restart && service avid-icps-manager restart`**
  - On a cluster. from any node: **`crm resource restart AvidAllEverywhere && crm resource restart AvidICPSEverywhere`**

Daily Maintenance

The following procedures should be completed on a daily basis and should take approximately 2 minutes. These steps can be completed on a “live”, in-production system.

Complete the following steps for a single-server configuration:

1. Log in to Linux as the “root” user.
2. Use the avid-ics script to verify the status of the MediaCentral services:

   ```
   avid-ics status
   ```

3. Examine the results to confirm that everything is listed as either “`[ OK ]`” or “`is running`”. Unless it is an expected status, any item that is listed as “`[ FAILED ]`”, “`is stopped`”, or “`not started`” must be investigated.

   For more information, see “Using the avid-ics Utility Script” on page 60.

Complete the following steps for a cluster configuration:

1. Log in to Linux as the “root” user on the cluster master node.
2. Run the “**`crm_mon -f1`**” command to display the cluster resources and status.

   The “`-f1`” switch tells Cluster Resource Monitor to display the output with failures only once.

3. Verify that there are no messages or errors under “Migration summary”. A healthy cluster should be free of messages and errors.

   For more information, see “Interpreting Failures in the Cluster” on page 86.

Weekly Maintenance

The following procedures should be completed on a weekly basis and should take approximately 2 minutes per server or node. These steps can be completed on a “live”, in-production system.

Complete the following steps for a single-server configuration:

1. Complete all single-server Daily Maintenance procedures.
2. Run the “**`df -h`**” command to confirm that all expected partitions, volumes, and storage(s) are mounted and accessible.

   As you revisit this process over the course of multiple weeks, also monitor the “Use%” for all volumes for changes. Verify that no volumes near 100% and that no volume suddenly increases the Use%. The root (`/`) directory is of particular importance. Changes should generally be gradual and not drastic.

   For more information, see “Verifying System Mount Points” on page 119.

3. If running MCS v2.6 or later, verify the status of sharded MongoDB using the mongo-checker command

   ```
   mongo-checker check-shard-status -u=admin -p=AvidAdmin_123!
   ```

   For more information on the output of this command, see the “Working with Sharded Mongo” chapter of the *Avid MediaCentral Platform Services Installation and Configuration Guide*.

4. Check system memory usage with the “`free`” command:

   ```
   free -m
   ```

   MCS memory usage can vary from one installation to another based on the site’s workflow. As you revisit this process over the course of multiple weeks, monitor the memory usage to determine your sites average values.

   For more information, see “Investigating Memory Usage” on page 117.

5. Verify time synchronization with the NTP server.

   For more information, see “Troubleshooting Time Synchronization” on page 120.

Complete the following steps for a cluster configuration:

1. Complete all cluster Daily Maintenance procedures.
2. Run the "**df -h**" command to confirm that all expected partitions, volumes, and storage(s) are mounted and accessible.

   This command should be run on each cluster node.

   As you revisit this process over the course of multiple weeks, also monitor the "Use%" for all volumes for changes. Verify that no volumes near 100% and that no volume suddenly increases the Use%. The root (/), DRBD (/mnt/drbd), and Gluster (/cache/) directories are of particular importance. Changes should generally be gradual and not drastic.

   For more information, see "Verifying System Mount Points" on page 119.
3. Run the "**drbd-overview**" command on the master and slave nodes to confirm their Connected status. The master nodes should be listed as Primary and the slave as Secondary.

   For more information, see "Verifying the DRBD Status" on page 78.
4. Check the DRBD volume's "max mount count" on the master node by running the following command:

   ```
   tune2fs -l /dev/drbd1 | grep -E 'Mount|Max'
   ```

   If the "Mount count" is greater than the "Maximum mount count", see the related article on the Avid Knowledge Base for more information and a process to resolve the issue.
5. If running MCS v2.6 or later, verify the status of sharded MongoDB using the mongo-checker command

   ```
   mongo-checker check-shard-status -u=admin -p=AvidAdmin_123!
   ```

   For more information on the output of this command, see the "Working with Sharded Mongo" chapter of the *Avid MediaCentral Platform Services Installation and Configuration Guide*.
6. Check system memory usage with the "free" command:

   ```
   free -m
   ```

   MCS memory usage can vary from one installation to another based on the site's workflow. As you revisit this process over the course of multiple weeks, monitor the memory usage to determine your sites average values.

   For more information, see "Investigating Memory Usage" on page 117.
7. Verify time synchronization with the NTP server.

   For more information, see "Troubleshooting Time Synchronization" on page 120.

Monthly Maintenance

The following procedures should be completed on a monthly basis and should take approximately 5 minutes per server. These steps can be completed on a "live", in-production system.

Throughout the process, watch for any warning messages or errors. If any issues are encountered, investigate and resolve them prior to releasing the system for use. If needed, contact Avid Customer Care at 800-800-AVID (2843) for assistance.

There are no monthly maintenance recommendations for a single-server configuration at this time.

Complete the following steps for a cluster configuration:

1. Complete all cluster Daily Maintenance and Weekly Maintenance procedures.
2. Run the "**rabbitmqctl cluster_status**" command to verify the status of the RabbitMQ cluster. Repeat this command on all cluster nodes.

   For more information, see "Verifying the Status of RabbitMQ" on page 76.
3. Run the "**acs-query**" command to check the health of the ACS bus. Repeat this command on all cluster nodes.

   For more information, see "Verifying ACS Bus Functionality" on page 75.
4. If your system has been configured with Gluster for cache volume replication, verify that all Gluster peers are known:

   ```
   gluster peer status
   ```

   A two node-cluster will report information similar to the following:

   ```
   [root@wavd-mcs01 ~]# gluster peer status
   Number of Peers: 1
   Hostname: wavd-mcs02
   Uuid: e54a6d62-fhhb-421b-b44f-33a1e2g7a297
   State: Peer in Cluster (Connected)
   ```

   Confirm that all nodes, excluding the local node, are "(Connected)".
5. If your system has been configured with Gluster for cache volume replication, use the following command to verify that all Gluster volumes or "bricks" are mounted on each of the cluster nodes:

   ```
   gluster volume info
   ```

   The following is an example of the "gl-cache-dl" volume in a two-node cluster:

   ```
   Volume Name: gl-cache-dl
   Type: Replicate
   Volume ID: 159e1427-6bba-4956-92b0-c1e54a377793
   Status: Started
   Number of Bricks: 1 x 2 = 2
   Transport-type: tcp
   Bricks:
   Brick1: wavd-mcs01:/cache/gluster/gluster_data_download
   Brick2: wavd-mcs02:/cache/gluster/gluster_data_download
   Options Reconfigured:
   storage.owner-uid: 497
   storage.owner-gid: 497
   ```

   Each cluster node should be listed with a Brick# entry under the "Bricks" section.
6. Manually verify that Gluster is replicating data across the cluster nodes.

   a. From any cluster node, create test files on the following Gluster shares:

   ```
   touch /cache/download/test001.txt
   touch /cache/fl_cache/test002.txt
   ```

b. Verify that the files created on your local system are replicated to all other cluster nodes. This can be accomplished by either opening an SSH session to each cluster node or you can use the Linux `ssh` command to verify the file replication from your current node:

```
ssh root@<node> ls <folder_path>
```

Where <*node*> is the hostname of the remote server and <*folder_path*> is the location of the test file.

You might be prompted to confirm that you wish to connect to the remote system. Enter “`yes`” to continue. You will also be prompted for the “root” user password of the remote system.

c. Once you have verified that file replication is functioning normally, remove the test files from the /cache directory:

```
rm /cache/download/test001.txt && rm /cache/fl_cache/test002.txt
```

You will be asked to confirm that you with to remove the files. Type: `yes`

The local and replicated copies of the files are deleted.

Common Troubleshooting Commands

The following table lists some helpful commands for general troubleshooting:

Command	Description
`acs-query`	Tests the RabbitMQ message bus and the avid-acs-ctrl-core service. With MediaCentral 2.5 and later, this command also tests the avid-acs-gateway service.
`avid-db dumpall`	Backs up the MCS databases
`avid-ics status`	Provides a status on many important services
`corosync-cfgtool -s` (cluster only)	Returns the IP and other stats for the node on which you issue the command.

`corosync-objctl`
or
`corosync-objctl | grep member`
(cluster only)
	Provides information about the corosync cluster.
	Returns the IP addresses of all nodes in the cluster. Each node should appear as “joined” as in the following example:
	`runtime.totem.pg.mrp.srp.members.892496394.ip=r(0)`
	`ip(192.168.10.51)`
	`runtime.totem.pg.mrp.srp.members.892496394.join_count=1`
	`runtime.totem.pg.mrp.srp.members.892496394.status=joined`
	`runtime.totem.pg.mrp.srp.members.909273610.ip=r(0)`
	`ip(192.168.10.52)`
	`runtime.totem.pg.mrp.srp.members.909273610.join_count=2`
	`runtime.totem.pg.mrp.srp.members.909273610.status=joined`
	`runtime.totem.pg.mrp.srp.members.993159690.ip=r(0)`
	`ip(192.168.10.53)`
	`runtime.totem.pg.mrp.srp.members.993159690.join_count=1`
	`runtime.totem.pg.mrp.srp.members.993159690.status=joined`

Command	Description
`crm` (cluster only)	Launches the Pacemaker Cluster Resource Manager in a shell mode. Once in the crm shell, tab twice for a list of options at each level. Type `help` for a list of commands. Press `q` to exit the help file. Hit CTRL-C on a Windows keyboard to exit the crm shell.
`crm_mon [-f]` (cluster only)	Opens the Pacemaker Cluster Resource Monitor. The -f option displays the fail-count for all services managed by Pacemaker.

`dmidecode | grep -A2 '^System Information'`	If you are accessing the server from a remote SSH session, this command prints the server information to the screen. Example:
	`System Information`
	`        Manufacturer: HP`
	`        Product Name: ProLiant DL360p Gen8`

`watch 'crm_mon -f1 | grep -A100 "Migration summary"'` (cluster only)	Depending upon your configuration and the number of managed resources, it can be difficult to see all messages related to the cluster when using the `crm_mon -f` command. This `watch` command provides a live status of the last 100 lines of the output of `crm_mon` following the “Migration summary”. The 100 value can be increased or decreased as desired.
`drbd-overview` (cluster only)	Prints DRBD status information to the screen. This information can also be obtained through the following command: service drbd status.
`echo 3 > /proc/sys/vm/drop_caches`	Clears the operating system’s memory cache. As of MCS v2.7, this command is run automatically as an daily cron job. Sites that are heavy users of MCS and see low free memory by the end of the day could benefit from moving this to an hourly cron job.

`gluster`
(cluster only)
	Queries GlusterFS peers. e.g.
	`gluster peer [`*command*`]`
	`gluster peer probe`

`history`	Prints a list of recently issued commands to the screen.
`ics_version`	Prints MCS version information to the screen.
`mongo-checker check-shard-status`	Displays the status of the sharded Mongo configuration in MCS v2.6 and later. For more information, see “Working with Sharded Mongo” in the *Avid MediaCentral Platform Services Installation and Configuration Guide*.

<table>
<tr><th>Command</th><th>Description</th></tr>
<tr><td>ping -c &lt;#&gt; <i>&lt;hostname or IP address&gt;</i></td><td>Verifies the connection to a remote system through a network “ping” request. The -c option defines the number of times the ping is sent.
When troubleshooting network issues, it might be useful to add a time stamp to the ping request so that the ping can be compared against log files. That command looks like the following:
ping <i>&lt;hostname or IP address&gt;</i> | perl -nle 'print scalar(localtime), " ", $_'
To send the output of a time-stamped ping request to a file, use the following command:
ping <i>&lt;hostname or IP address&gt;</i> | perl -nle 'BEGIN {$|++} print scalar(localtime), " ", $_' &gt; <i>&lt;file_name&gt;</i>
Press CTRL-C to stop the command and close the file.
For more information on the use of ping, see “Verifying Network Connectivity” on page 69.</td></tr>
<tr><td>ps -ae | grep [-c] intern</td><td>This command polls the max-edit player and returns information regarding the connections to the player on the current server. Example:
[root@wavd-mcs01]# ps -ae|grep intern
105036 pts/0   00:00:49 max-edit-intern
If you have a cluster with multiple servers, this command can help you verify which node is servicing a playback request.
Adding the [-c] parameter provides a count of playback streams being serviced by the current server.</td></tr>
<tr><td>service avid-all clear-cache</td><td>This command deletes the /cache/mob-fetch content which clears the cache of the player service. This command would need to be run on all servers in a cluster configuration. This command could be useful if troubleshooting Media Offline issues.</td></tr>
<tr><td>service avid-all env</td><td>Displays the information entered in the MediaCentral UX System Settings.</td></tr>
<tr><td>system-backup [-b | -r]</td><td>Often run prior to an MCS upgrade, this script backs up the system settings and MCS databases. When run with the -r option, the script restores the backed-up data.
For more information, see “Appendix E” of the <i>MediaCentral Platform Services Upgrade Guide</i>.</td></tr>
</table>

Investigating Memory Usage

Administrators can check the amount of used and available memory on a server through the Linux “`free`” command. By default, the command shows available RAM in kilobytes, but a `-m` or `-g` switch can be added to show the values in megabytes or gigabytes, respectively.

Below is an example of a system with 96 GB (96727 shown) of RAM:

```
[root@wavd-mcs01 ~]# free -m
             total       used        free     shared     buffers      cached
Mem:         96727       6565       90162          0          76       1092
-/+ buffers/cache:       5396       91331
Swap:         4095          0        4095
```

Take note of the total, used and free values. Once the “used” value surpasses the “free” value, the “Swap” memory will begin to be consumed. The “swap” is a portion of hard drive space that is used as virtual memory. It is slower and less efficient than actual RAM and should generally be avoided if possible.

If you notice that the used memory is reaching the total memory value or if the system has begun to use the swap space, it is recommended to reboot the server to flush the memory. If you are running an older configuration with 64 GB of RAM, you might want to consider upgrading the amount of installed memory to 96 or 128 GB of RAM.

Identifying System Hardware

Linux provides a variety of tools and commands for determining the server components without opening the server at a hardware level.

Determining the CPU Type

The CPU can be easily determined through the following command:

```
cat /proc/cpuinfo | grep "model name"
```

The output of this command displays multiple lines indicating the total number of CPU cores available on the processors. If hyper-threading is enabled, additional cores might be listed. For instance when the command is issued on a system equipped with two 10-core, Intel Xeon E5-2670 v2 processors, forty lines are printed to the screen. For example:

```
model name      : Intel(R) Xeon(R) CPU E5-2670 v2 @ 2.50GHz
```

To get a quick count on the total number of cores, issue the following command:

```
cat /proc/cpuinfo | grep processor | wc -l
```

Determining Installed Memory

MediaCentral systems running on older hardware might have less RAM than is currently recommended for new installations. If client load increases or new features are added, such as Media Index, you might be required to upgrade the amount of memory in your system. The Linux `dmidecode` command can assist you in determining how much RAM is currently installed as well as the maximum amount of RAM supported in the server hardware.

See the following example output:

```
[root@wavd-mcs01 ~]# dmidecode -t 16
```

```
# dmidecode 2.11
SMBIOS 2.8 present.
# SMBIOS implementations newer than version 2.7 are not
# fully supported by this version of dmidecode.

Handle 0x1000, DMI type 16, 23 bytes
Physical Memory Array
    Location: System Board Or Motherboard
    Use: System Memory
    Error Correction Type: Single-bit ECC
    Maximum Capacity: 384 GB
    Error Information Handle: Not Provided
    Number Of Devices: 12

Handle 0x1001, DMI type 16, 23 bytes
Physical Memory Array
    Location: System Board Or Motherboard
    Use: System Memory
    Error Correction Type: Single-bit ECC
    Maximum Capacity: 384 GB
    Error Information Handle: Not Provided
    Number Of Devices: 12
```

The server in the example above has 24 total slots which support a maximum of 768 GB of RAM. These numbers are determined by examining the Number of Devices as well as the Maximum Capacity for each CPU.

When used with the `memory` switch, the `dmidecode` command can be used to provide detailed information on each DIMM. The following is a shorted example showing one DIMM:

```
[root@wavd-mcs01 ~]# dmidecode -t memory
```

```
Handle 0x1100, DMI type 17, 40 bytes
Memory Device
    Array Handle: 0x1000
Error Information Handle: Not Provided
Total Width: 72 bits
Data Width: 64 bits
Size: 16384 MB
Form Factor: DIMM
Set: None
Locator: PROC  1 DIMM  1
Bank Locator: Not Specified
Type: DDR3
Type Detail: Synchronous Registered (Buffered)
Speed: 1866 MHz
Manufacturer: HP
Serial Number: Not Specified
Asset Tag: Not Specified
Part Number: 712383-081
Rank: 2
Configured Clock Speed: 1866 MHz
```

The same command can be used with a modifier to determine where the DIMMs are installed in the system as shown in the following example:

```
[root@wavd-mcs01 ~]# dmidecode -t memory | grep Size

    Size: 16384 MB
    Size: No Module Installed
    Size: No Module Installed
    Size: 16384 MB
    Size: No Module Installed
    Size: No Module Installed
    Size: No Module Installed
    Size: No Module Installed
    Size: 16384 MB
    Size: No Module Installed
    Size: No Module Installed
    Size: 16384 MB
    Size: 16384 MB
    Size: No Module Installed
    Size: No Module Installed
    Size: 16384 MB
    Size: No Module Installed
    Size: No Module Installed
    Size: No Module Installed
    Size: No Module Installed
    Size: 16384 MB
    Size: No Module Installed
    Size: No Module Installed
    Size: 16384 MB
```

The output of this command shows the 24 memory slots; populated with eight 16 GB DIMMs.

Any RAM that is added to the system is automatically recognized by RHEL and is available for use with MediaCentral. No additional software configuration is required to access the increased RAM.

For more information on memory configurations, see the MediaCentral Platform Services Hardware Guide.

Verifying System Mount Points

The Linux “df -h” command can be used to confirm that all expected partitions, volumes and storage(s) are mounted and accessible to the operating system and to MediaCentral.

At the Linux command prompt, type: **df -h**

On a single server, the output should look similar to the following:

```
[root@wavd-doc01 ~]# df -h
    Filesystem                         Size  Used Avail Use% Mounted on
    /dev/mapper/lvg-lv_root            93G    14G   74G  16% /
    tmpfs                              1.9G     0  1.9G   0% /dev/shm
    /dev/sda1                          504M   45M  434M  10% /boot
    /dev/mapper/vg_ics_cache-lv_ics_cache  99G   189M   94G   1% /cache
    wavd-isis                          15T   2.9T   12T  20% /mnt/ICS_Avid_Isis/wavd-isis
```

On the master cluster node, the output should look similar to the following:

```
[root@wavd-mcs01 ~]# df -h
    Filesystem                            Size  Used Avail Use% Mounted on
    /dev/mapper/lvg-lv_root                75G   15G   57G  21% /
    tmpfs                                 1.9G   46M  1.9G   3% /dev/shm
    /dev/sda1                             504M   45M  434M  10% /boot
    /dev/mapper/vg_ics_cache-lv_ics_cache  99G  189M   94G   1% /cache
    wavd-mcs01:/gl-cache-dl                99G  189M   94G   1% /cache/download
    wavd-mcs01:/gl-cache-fl                99G  189M   94G   1% /cache/fl_cache
    wavd-mcs01:/gl-cache-mcam              99G  189M   94G   1% /cache/render
    /dev/drbd1                             20G  3.5G   16G  19% /mnt/drbd
    wavd-isis                              15T  2.9T   12T  20% /mnt/ICS_Avid_Isis/wavd-isis
```

Verify that the expected partitions are mounted on the MCS server(s). Expected partitions may include:

- The `/cache` partition used to stream media through the player.
- Avid shared storage workspace(s). In the above example, an ISIS system is mounted as `/mnt/ICS_Avid_Isis/wavd-isis`
- Shared storage location for Interplay MAM workflows (not shown above).
- (cluster only) Gluster volumes for media replication between the cluster nodes.
- (cluster only) The drbd database: `/mnt/drbd`. This mount will only exist on the master node in a cluster.

Troubleshooting Time Synchronization

The MediaCentral server clock is synchronized with the “house” time through the network time protocol (NTP) daemon, ntpd. During the initial installation and configuration of the MCS servers, a Linux cron job is created on each server to ensure that time-sync is continuously maintained. Time synchronization is critically important on both single-server and clustered configurations.

If the MCS servers are not in sync with the house clock, any number of following issues could occur:

- Media Offline messages in the MCUX player due to communication issues with the Interplay Production Media Indexer
- Slow or inconsistent connections to the Interplay Production Engine
- MCS cluster failures
- Errors and failures when issuing Send to Playback requests
- And more...

To verify the system date and time:

The current date and time can be verified using the Linux `date` command. For example:

```
[root@wavd-mcs01 ~]# date
Mon Feb 22 14:41:04 EST 2016
```

The local date and time can be compared against the configured NTP server using the `ntpdate` command. For example:

```
ntpdate -q <ntp_server>
```

The following example shows a negative 62 second offset between the local time and the NTP server clock:

```
[root@wavd-mcs01 ~]# ntpdate -q 192.168.10.25
server 192.168.10.25, stratum 3, offset -62.400742, delay 0.02573
22 Feb 14:44:44 ntpdate[89458]: step time server 192.168.10.25 offset -
62.400742 sec
```

To update the system date and time:

If you find a small offset as in the example above, the following command can be used to resync the local clock with the NTP server:

```
/usr/sbin/ntpd -q -u ntp:ntp
```

If the above command returns no information, it could be that the ntpd service is currently running. Manual command line interaction requires the service to be stopped.

The following example indicates that the local clock has corrected a -62 second offset:

```
[root@wavd-mcs01 ~]# /usr/sbin/ntpd -q -u ntp:ntp
ntpd: time set -62.401179s
```

If the time offset is too great, the above command will not update the clock. Instead, use the following command to “force” the clock update:

```
ntpdate <ntp_server>
```

For more information on time synchronization and NTP, see the following resources:

- “Verifying Time Synchronization” on page 75 of this document.
- “Configure Date and Time Settings” in the *MediaCentral Platform Services Installation and Configuration Guide*.
- “Time Synchronization for Avid Interplay systems” on the Avid Knowledge Base.

Responding to Automated Cluster E-mail

In cluster configurations, Pacemaker is configured by default to send automated e-mails to notify administrators of important events. The following table presents the e-mail types that can be sent and the remedial action needed.

E-mail Type	Description	Action Needed
Node Down/ Removed from Cluster	• Indicates that corosync has stopped or unexpectedly crashed.	As long as the activity is expected, no action is required. If this is an unexpected alert, check power and networking cables for loose connections. Verify that the node is accessible through direct keyboard input or an SSH connection.
Node Up /Joined Cluster	• During installation, a new node has successfully joined the cluster.	None.
DRBD Split Brain	• DRBD is operating independently on the two nodes where it is running.	The cluster requires immediate attention to remedy the situation. To remedy, wipe out the DRBD database on one of the nodes, then rejoin that node to the DRBD primary node. See “Correcting a DRBD Split Brain” on page 131.
DRBD Split Brain Recovery	• DRBD has been successfully reconfigured.	None.
Peer Connection lost	• DRBD node is put into Standby mode. • DRBD has lost connection to its peer.	If the DRBD node is put into Standby, this message is expected and no action is required. If this is an unexpected alert, the issue should be investigated further. For more information, see “Verifying the DRBD Status” on page 78.
Peer Connection established	• DRBD node is put back online, after having entered Standby mode. • DRBD has reconnected to its peer.	

Troubleshooting RabbitMQ

The Avid Knowledge Base includes a page that provides detailed instructions on reviewing the status of RabbitMQ and troubleshooting any related errors. See the following link for details:

http://avid.force.com/pkb/articles/en_US/troubleshooting/RabbitMQ-cluster-troubleshooting

Verifying the RabbitMQ Status

In addition to the information provided on the Avid Knowledge Base, the acs-broker-status command can be used to quickly return the status of RabbitMQ related components.

To check the status of RabbitMQ, enter the following on any of the cluster nodes:

```
acs-broker-status
```

You should see an output similar to the following:

```
[root@wavd-mcs01 ~]# acs-broker-status
Checking Broker Configuration...
Querying data from rabbitmqctl...
High Watermarks:
        memory                                             [  OK  ]
        disk                                               [  OK  ]
Cluster:
        network partion                                    [  OK  ]
VHosts:
        acs present                                        [  OK  ]
Users:
        acs_admin present                                  [  OK  ]
        acs_admin is an administrator                      [  OK  ]
        acs_admin permissions correct                      [  OK  ]
        acs_user present                                   [  OK  ]
        acs_user is an administrator                       [  OK  ]
        acs_user permissions correct                       [  OK  ]
        guest is not present                               [  OK  ]
Exchanges:
        Local.Requests present                             [  OK  ]
        Local.Broadcasts present                           [  OK  ]
        Local.Channels present                             [  OK  ]
        Zone.Requests present                              [  OK  ]
        Zone.Broadcasts present                            [  OK  ]
        Zone.Channels present                              [  OK  ]
        Zone.Replies present                               [  OK  ]
        MultiZone.Channels present                         [  OK  ]
        MultiZone.Broadcasts present                       [  OK  ]
        Fanout.Channels present                            [  OK  ]
        Fanout.Broadcasts present                          [  OK  ]
Bindings:
        Fanout.Broadcasts -> Local.Broadcasts              [  OK  ]
        Fanout.Broadcasts -> MultiZone.Broadcasts          [  OK  ]
        Fanout.Channels -> Local.Channels                  [  OK  ]
        Fanout.Channels -> MultiZone.Channels              [  OK  ]
```

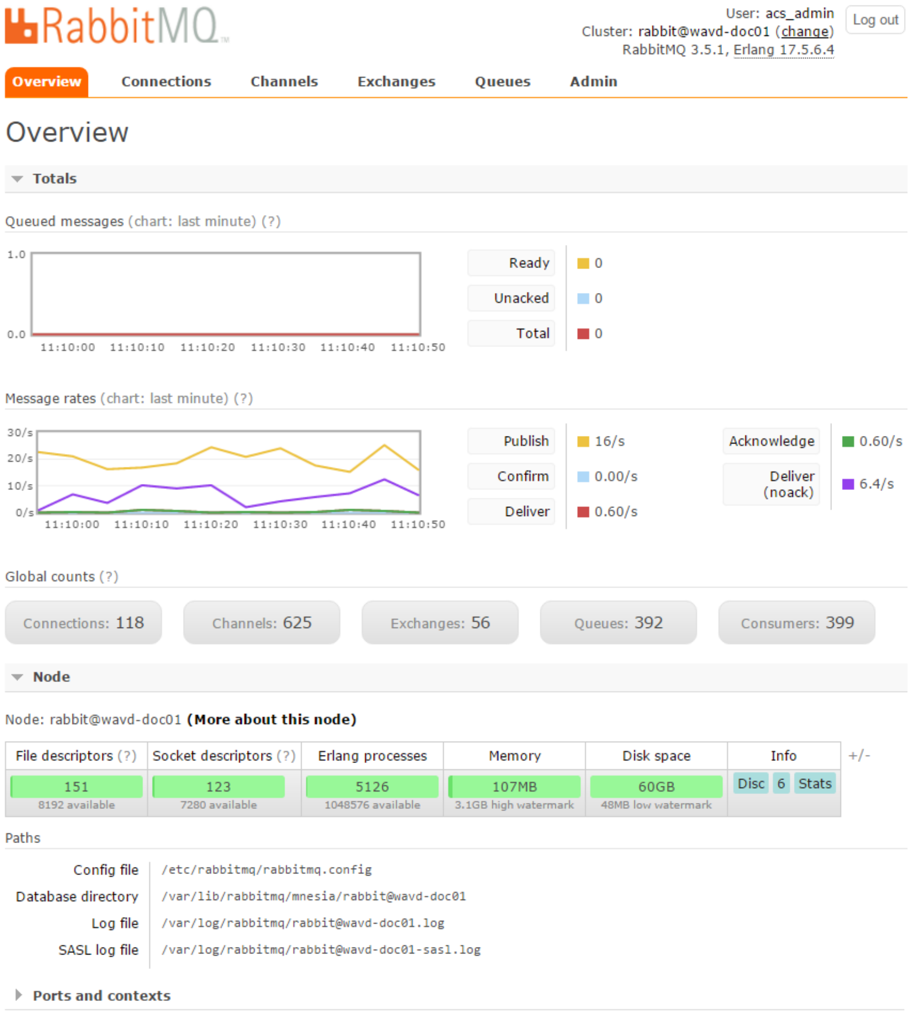
An “OK” response indicates that the acs-broker and RabbitMQ communication is normal.

RabbitMQ Web Management Utility

RabbitMQ offers a utility that provides detailed information on processes, queues and overall system status, including information on all nodes in a cluster configuration.

To access this utility, enter the following in a web browser: `http://<hostname>:15672`

When presented with the login page, enter the following user / password: acs_admin / cl0ud



User: acs_admin
Cluster: rabbit@wavd-doc01 (change)
RabbitMQ 3.5.1, Erlang 17.5.6.4

Overview

Totals

Queued messages (chart: last minute) (?)

Ready	🟨 0
Unacked	🟦 0
Total	🟥 0

Message rates (chart: last minute) (?)

Publish	🟨 16/s	Acknowledge	🟩 0.60/s
Confirm	🟦 0.00/s	Deliver (noack)	🟪 6.4/s
Deliver	🟥 0.60/s		

Global counts (?)

Connections: 118 Channels: 625 Exchanges: 56 Queues: 392 Consumers: 399

Node

Node: rabbit@wavd-doc01 **(More about this node)**

File descriptors (?)	Socket descriptors (?)	Erlang processes	Memory	Disk space	Info
151 8192 available	123 7280 available	5126 1048576 available	107MB 3.1GB high watermark	60GB 48MB low watermark	Disc 6 Stats

Paths

Config file	/etc/rabbitmq/rabbitmq.config
Database directory	/var/lib/rabbitmq/mnesia/rabbit@wavd-doc01
Log file	/var/log/rabbitmq/rabbit@wavd-doc01.log
SASL log file	/var/log/rabbitmq/rabbit@wavd-doc01-sasl.log

Notice that the page is divided into multiple tabs that are listed across the top of the utility:

- **Overview -** Provides general information about the status of RabbitMQ. If you have a RabbitMQ cluster, each node would be listed separately under the Node category.
- **Connections -** Represent TCP connections between a host and the RabbitMQ server.
- **Channels -** A channel is a virtual connection inside a connection.
- **Exchanges -** Exchanges process messages between systems and queues.
- **Queues -** As previously described, a "queue" is essentially a container created within the RabbitMQ message broker responsible for storing and processing "messages" between connected systems.

Each queue must have an associated consumer. If a queue does not have a consumer, messages can begin to accumulate which would lead to an alarm. If left unchecked, the messages would eventually consume all system memory and cause a system crash. You can verify if there are any unacknowledged queues by sorting the display by the "Unacked" column. If everything is working normally, all queues should list zero unacknowledged messages.

If there are any queues that do not have a consumer, see "Cleaning Stale Queues" on page 125 for information on how to resolve this issue.

- **Admin** - The first page of the Admin tab shows the two users created by the Avid installation scripts: acs_admin and acs_user. These users must not be altered.

  If you have a multi-zone configuration, you can click on the "Federation Upstreams" link on the right and see each of the connected zones.

For more information about the web management utility, see https://www.rabbitmq.com/management.html.

Cleaning Stale Queues

If a queue does not have a consumer, messages can begin to accumulate which would lead to an alarm. If left unchecked, the messages would eventually consume all system memory and cause a system crash. In addition to the RabbitMQ Web Management Utility described above, the `acs-clean` command can also be used to verify (and clean) unacknowledged queues.

Although you may obtain similar "stale queue" results in releases prior to v2.5, this process only applies to MediaCentral Platform Services v2.5 and later.

To check for unacknowledged queues using acs-clean:

- Enter the following command on a single server or a cluster master node:

  ```
  /opt/avid/bin/acs-clean -c -m '.*\.requests$'
  ```

  A healthy system with no unacknowledged queues replies with output similar to the following:

  ```
  localhost: GET /api/queues/acs -> 200
  No queues to display.
  ```

  Alternatively, a system with unacknowledged queues might show output similar to the following:

  ```
  localhost: GET /api/queues/acs -> 200
  âââââââââââââââââââââââââââââââââââââââââââââââââââââââââ¬ââââââââââ¬âââââââââââ
  â Queue                                                 â Consumers â Messages â
  â com.avid.cc.conversion.com.avid.central.cc.Conversioâ¦ â 0         â 0       â
  âââââââââââââââââââââââââââââââââââââââââââââââââââââââââ´ââââââââââ´âââââââââââ
  Found 1 queues
  ```

  If any queues are found, proceed to either the single server or cluster process found on the next page to clean the stale queues on the server.

- For more information about the options used with this command, use the help option:

  ```
  /opt/avid/bin/acs-clean --help
  ```

To clean the RabbitMQ queue on a single server:

1. Stop the MediaCentral services to eliminate the creation of new requests to RabbitMQ:

   ```
   avid-ics stop
   ```

2. Enter the following command on the MCS server:

   ```
   /opt/avid/bin/acs-clean -c -d -f -m '.*\.requests$'
   ```

When the cleanup occurs, you might see more queues being removed than what were listed in the verification command. This is normal.

3. Once the process has completed, verify that the unacknowledged queues have been cleaned:

   ```
   /opt/avid/bin/acs-clean -c -m '.*\.requests$'
   ```

   A healthy system with no unacknowledged queues replies with output similar to the following:

   ```
   localhost: GET /api/queues/acs -> 200
   No queues to display.
   ```

4. Restart the MediaCentral services:

   ```
   avid-ics start
   ```

To clean the RabbitMQ queue on a cluster:

1. Before cleaning the queues, the Corosync cluster must first be placed in standby mode:

   a. Begin by putting all Corosync load-balancing nodes and the slave node into standby. Complete this process one node at a time, waiting 30 seconds between each node:

      ```
      crm node standby <node hostname>
      ```

   b. Wait 30 seconds and put the Corosync master node into standby:

      ```
      crm node standby <node hostname>
      ```

   c. Open the Cluster Resource Monitor on any node to verify that all nodes are in standby:

      ```
      crm_mon
      ```

      Press CTRL-C on a Windows keyboard or CMD-C on a Mac keyboard to exit the console.

2. It is imperative that rabbitmq-server service is running on all nodes. Verify the status of the service with the following command:

   ```
   service rabbitmq-server status
   ```

   Repeat the above command on each node.

   If the service is not running on any node, start the service:

   ```
   service rabbitmq-server start
   ```

3. Delete the stale queues by entering the following command from any node:

   ```
   /opt/avid/bin/acs-clean -c -d -f -m '.*\.requests$'
   ```

When the cleanup occurs, you might see more queues being removed than those listed in the verification command. This is normal.

4. Once the process has completed, verify that the unacknowledged queues have been cleaned:

   ```
   /opt/avid/bin/acs-clean -c -m '.*\.requests$'
   ```

A healthy system with no unacknowledged queues replies with output similar to the following:

```
localhost: GET /api/queues/acs -> 200
No queues to display.
```

5. Bring the cluster master node back online:

   `crm node online <master node hostname>`

   Use the Cluster Resource Monitor, `crm_mon`, to verify that all resources start normally.

6. Bring the slave node and any load balancing nodes online:

   `crm node online <node hostname>`

   Repeat the above command for each node. This step can be completed on all remaining nodes simultaneously.

7. Use the Cluster Resource Monitor to verify that all nodes appear online:

   `crm_mon -f`

   If there are fail counts listed, run the Cluster Resource Manager cleanup command to reset them:

   `crm resource cleanup <rsc> [<node>]`

   <*rsc*> is the resource name of interest: AvidIPC, AvidUMS, AvidACS, pgsqlDB (or another)

   <*node*> (optional) is the node of interest.

Troubleshooting DRBD

Recall that DRBD runs on the master and slave nodes only, and is responsible for mirroring the contents of a partition between master and slave. The mirrored partition is used to store several databases used by MediaCentral. For details, see "DRBD and Database Replication" on page 46. This section does not apply to single-server configurations.

This section presents common DRBD problems and solutions. Typical problems in DRBD include:

- A lack of primary-secondary connectivity
- The secondary operating in standalone mode
- Both nodes reporting connectivity but neither one in the role of master
- Both nodes reporting themselves in the role of master

For more information on troubleshooting DRBD issues, see the related article on the Avid Knowledge Base.

Verify the DRBD Status

The following command is used to verify that DRBD is operating normally on the master and slave nodes:

```
drbd-overview
```

When run on the master node, the output should look like the following:

```
1:r0/0 Connected Primary/Secondary UpToDate/UpToDate C r----- /mnt/drbd ...
```

When run on the slave node, the output should look like the following:

```
1:r0/0 Connected Secondary/Primary UpToDate/UpToDate C r-----
```

The following sections are examples of issues found with DRBD and how to resolve them.

Master Node: WFConnection

```
1:r0/0 WFConnection Primary/Unknown UpToDate/DUnknown C r----- /mnt/drbd ext4
20G 397M 18G 3%
```

Summary: The DRBD master node cannot connect to the DRBD slave node:

`WFConnection`	The master node is waiting for a connection from the slave node (i.e. the slave node cannot be found on the network).
`Primary/Unknown`	This node is the master, but the slave node cannot be reached.
`UpToDate/DUnknown`	The database on the master is up to date, but the state of the database on the slave node is not known.

Action Required: Make the connection manually. Refer to the instructions in "Manually Connecting the DRBD Slave to the Master" on page 130.

If the master node reports WFConnection while the slave node reports StandAlone, it indicates a DRBD split brain. See "Correcting a DRBD Split Brain" on page 131 for additional details.

Slave Node: Standalone

```
1:r0/0 StandAlone Secondary/Unknown UpToDate/DUnknown r-----
```

Summary: The slave cannot connect to the master.

`StandAlone`	The slave node is operating on its own. (StandAlone)
`Secondary/Unknown`	The slave node is the secondary, but the primary cannot be found (Secondary/Unknown)
`UpToDate/DUnknown`	The database on the slave node is up to date, but the state of the database on the master is unknown (UpToDate/DUnknown)

Action Required: Make the connection manually. Refer to the instructions in "Manually Connecting the DRBD Slave to the Master" on page 130.

If the master node reports WFConnection while the slave node reports StandAlone — it indicates a DRBD split brain. See "Correcting a DRBD Split Brain" on page 131 for addtional details.

Both Nodes: Secondary/Secondary

```
1:r0/0 Connected Secondary/Secondary UpToDate/UpToDate C r-----
```

Summary: The nodes are connected, but neither is master.

`Connected`	A connection is established.
`Secondary/Secondary`	Both nodes are operating as the slave node. That is, each is acting as the peer that receives updates.
`UpToDate/Unknown`	The database on the local node is up to date, but the state of the database on the remote node is not known.

Action needed: This usually indicates a failure within the Pacemaker PostgreSQL resource group. For example, if Pacemaker cannot mount the DRBD device as a file system, DRBD will start successfully, but writing data to disk and database replication cannot take place.

To investigate the issue further:

1. Use the Pacemaker Cluster Resource Monitor to verify if all services are running.

   ```
   crm_mon -f
   ```

   For details, see "Cluster Resource Monitor" on page 80.

2. Reset fail-counts.

   For details, see "Identifying Failures in CRM" on page 84.

3. Restart failed Pacemaker resources or the underlying Linux services.

4. If all services in the PostgreSQL resource group are operating as expected, the problem may lie at a deeper level of the Linux operating system.

   For details, see "Working with System Logs" on page 133.

   Solving this issue can be complex. If the above suggestions do not resolve the problem, consult your Avid representative for further troubleshooting.

Both Nodes: Standalone and Primary

```
1:r0/0 StandAlone Primary/Unknown UpToDate/Unknown C r----- /mnt/drbd ext4 20G
397M 18G 3%
```

```
1:r0/0 StandAlone Primary/Unknown UpToDate/Unknown C r-----
```

Summary: A DRBD “split brain” has occurred. Both nodes are operating independently, reporting themselves as the master node, and claiming their database is up to date.

`StandAlone` Each node believes it is the DRBD master. Each will operate independently until an administrator manually intervenes.

`Primary/Unknown` The local node believes it is the DRBD master. The remote node is not connected which results in the Unknown status.

The key indicator of this type of DRBD split brain is both nodes reporting themselves as the Primary.

`UpToDate/Unknown` The database on the local node is up to date, but the state of the database on the remote node is not known.

Action Needed: See “Correcting a DRBD Split Brain” on page 131 for more details.

Manually Connecting the DRBD Slave to the Master

When the master and slave nodes are not connecting automatically, you will have to make the connection manually. You do so by telling the slave node to connect to the resource owned by the master. The process below is only valid if the DRBD master node is in a WFConnection state. This section does not apply to single-server configurations.

To manually connect the DRBD slave to the master:

1. Log in to any node in the cluster as *root* and start the Cluster Resource Monitor: **`crm_mon`**
2. Identify the slave node by looking for the line containing “Master/Slave Set”. For example:

```
Master/Slave Set: ms_drbd_postgres [drbd_postgres]
     Masters: [ wavd-mcs01 ]
     Slaves:  [ wavd-mcs02 ]
```

In this situation, it is possible that the DRBD master may not be the same as the Pacemaker cluster master. Use the tools detailed in this document to identify the DBRB master node.

3. On the slave node run the following command:

```
drbdadm connect r0
```

4. Verify the reconnection was successful:

```
drbd-overview
```

The output on the master node should resemble the following:

```
1:r0/0 Connected Primary/Secondary UpToDate/UpToDate C r----- /mnt/drbd
ext4 20G 397M 18G 3%
```

The output on the slave node should resemble the following:

```
1:r0/0 Connected Secondary/Primary UpToDate/UpToDate C r-----
```

Correcting a DRBD Split Brain

A DRBD split brain describes the situation in which both DRBD nodes are operating completely independently. Further, there is no connection between them, hence data replication is not taking place. A DRBD split brain must be remedied as soon as possible as data can be easily lost due to the lack of replication between the nodes. This section does not apply to single-server configurations.

To recover from a split brain, you must force the MCS cluster master node to take on the role of DRBD master. You then discard the database associated with the DRBD slave node, and reconnect it to the established master.

Discarding the database on the slave node does not result in a full re-synchronization from master to slave. The slave node has its local modifications rolled back, and modifications made to the master are propagated to the slave.

To recover from a DRBD split brain:

1. Log in to any node in the cluster as *root* and start the Cluster Resource Monitor:

   ```
   crm_mon
   ```

2. Identify the master node by looking or the line containing “Master/Slave Set”. For example:

   ```
   Master/Slave Set: ms_drbd_postgres [drbd_postgres]
        Masters: [ wavd-mcs01 ]
        Slaves:  [ wavd-mcs02 ]
   ```

It is possible that you may not be able to identify the master node through the Cluster Resource Monitor when DRBD is running in a split brain state. In this event you must determine the master node using your best judgment.

3. On the master run the following command:

   ```
   drbdadm connect r0
   ```

   This ensures the master node is connected to the *r0* resource. This DRBD resource holds the databases. It was given the name *r0* during the initial DRBD creation process.

4. On the slave run the following command:
5. **drbdadm connect --discard-my-data r0**

   If the slave node is already in a “WFConnection” state, you will see the following message:

   ```
   Failure: (102) Local address (port) already in use.
   ```

   If you encounter this message, explicitly disconnect the slave node from the resource using the following command and then repeat the connect command:

   ```
   drbdadm disconnect r0
   ```

6. Verify the recovery was successful:

   ```
   drbd-overview
   ```

   The output on the master node should resemble the following:

   ```
   1:r0/0 Connected Primary/Secondary UpToDate/UpToDate C r----- /mnt/drbd
   ext4 20G 397M 18G 3%
   ```

   The output on the slave node should resemble the following:

   ```
   1:r0/0 Connected Secondary/Primary UpToDate/UpToDate C r-----
   ```

Monitoring Load Balancing

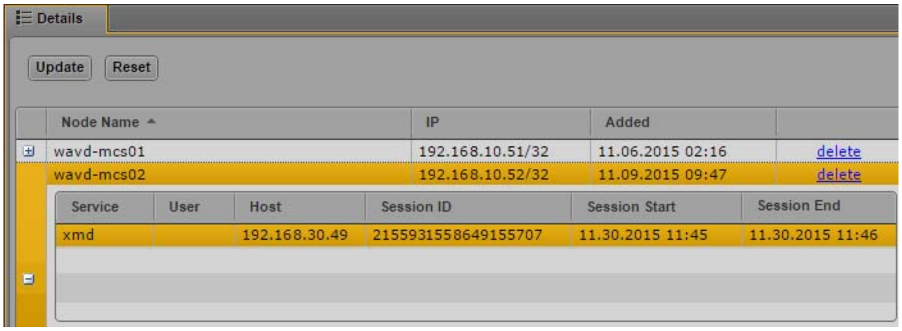
Incoming playback requests are routed to the cluster’s virtual IP address, and then distributed evenly throughout the cluster. Load balancing is automatic, and supports many concurrent clients. The load balancing daemon/service, *xmd*, runs multiple instances on each node in the cluster.

To monitor load balancing in an Interplay MAM deployment, a server hostname must be entered in the Player section of the MCPS settings. See “Configuring MCS for Interplay MAM” in the Avid MediaCentral Platform Services Installation and Configuration Guide.

To monitor load-balancing:

1. Sign in to MediaCentral UX as a user with administrator-level access.
2. Select System Settings from the Layout selector.
3. In the Settings pane, select MCPS > Load Balancer.

   A list of all known servers appears on the right side of the page:



	Node Name	IP	Added	
⊞	wavd-mcs01	192.168.10.51/32	11.06.2015 02:16	delete
⊟	wavd-mcs02	192.168.10.52/32	11.09.2015 09:47	delete

Service	User	Host	Session ID	Session Start	Session End
xmd		192.168.30.49	2155931558649155707	11.30.2015 11:45	11.30.2015 11:46

The following table explains the information:

Service	Description
Node Name	Host name of the load-balancing node. Click the plus (+) button to reveal details about a particular node, as explained below.
Service	The *xmd* service is the playback service responsible for delivering video from the MCS server to the player embedded in the web browser.
User	*Reserved for future use.*
Host	The IP address of the client machine (and end-user) to which video is being served.
Session ID	The session ID associated with the playback session.
Session Start	The time (MM.DD.YYYY HH:SS) at which the player embedded in the browser connected to the MCS server.
Session End	The time at which the session was terminated.
IP	The node’s IP address (e.g. XXX.XX.XX.XXX/32). Note that a /32 netmask indicates that point-to-point communication is used by the load balancing service. It does not reflect the facility netmask in use across the facility network.
Added	The time (MM.DD.YYYY HH) at which the node was added to the load-balancer.

4. The following table explains the possible actions you can take:

Action	Description
Update	Updates information (e.g. host, session ID, etc.) for all nodes registered for load balancing.
Reset	Flushes the database where nodes register for load balancing. Helpful if you have removed a node from the cluster, but it continues to appear in the list. Nodes capable of load balancing will self-register in a short time.
delete	Explicitly removes the named node from the load balancing database. This does not remove the node from the cluster.

Working with System Logs

MCS and its supporting services — such as Pacemaker, Corosync, and RabbitMQ — produce numerous logs. These are stored in the standard RHEL directory and subdirectories:

```
/var/log
```

Typically, log files are created in the following format:

```
<process>.log
```

For example:

```
spooler.log
spooler.log-201310.25.gz
spooler.log.old20131024_141055
```

Note the following:

- *.log are current log files, for the active process.
- *.gz are “rotated out” log files, compressed and with a date appended.
- *.old are backlogs.

Log files are *rotated* (replaced), compressed and eventually deleted automatically by the Linux *logrotate* log management utility. In addition, most MCS logs have the following default characteristics, determined by the *logrotate* configuration file (`/etc/logrotate.conf`**)**:

- Fresh logs are begun with each reboot
- New log files are uncompressed text files (some are binaries)
- Older logs are rotated (replaced) weekly
- Older logs are stored in the *gzip* format
- Four weeks worth of backlogs are kept
- A new empty log file is created after rotating out the old one
- Date is appended as suffix on the rotated file

Specific processes can override the logrotate configuration file settings by supplying their own configuration file in either the `/etc/logrotate.d` *or* `/etc/logrotate.hourly` *directories. If a log file is not behaving as expected, check there.*

Understanding Log Rotation and Compression

The Linux *logrotate* utility runs and compresses the old logs daily. Although it is invoked by the Linux *cron* daemon, the exact runtime for *logrotate* cannot be stated with accuracy. It varies, for example, depending on when the system was most recently rebooted, but it does not run at a fixed time after the reboot. This is by design, in order to vary and minimize the impact on other system resources. By default, rotated logs files are store as *gzip* (.gz) compressed files.

The production of logs is controlled by the following files:

- **/etc/cron.daily/logrotate** specifies the job to be run and the file containing configuration parameters
- **/usr/sbin/logrotate** is the job that is run
- **/etc/logrotate.conf** is the file containing configuration parameters
- **/etc/logrotate.d** is a directory containing additional configuration information that might override the default instructions
- **/etc/logrotate.hourly** is a directory introduced in MCS v2.8.0 that contains additional configuration information that might override the default instructions

Be aware that some systems utilize their own log management system and do not use the logrotate utility

Further details on the log rotation configuration files are beyond the scope of this document. For more information, see the Linux *man* page for *logrotate* by typing the following at the Linux command line: `man logrotate`

Log File Format

MediaCentral Platform Services v2.8 introduced a standardized format for all logs written to the `/log/avid/avid-interplay-central` directory. Each log entry adheres to the following format:

```
<timestamp> [<category>] [<component>] [<eventcode>] {<private tags>}
<message>
```

Log Entry	Description	Example
Time Stamp	Local time in 24 hour format	YYYY-MM-ddTHH:mm:ss.SSS±hh:mm
	Timezone based on UTC calculation with the format according to ISO 8601	Example: 2016-10-30T23:23:20.430+02:00
Category / Log Level	Log levels ranging from Informational to Emergency	See the table below for Log Level descriptions.
Component	This is the name of the component that has generated the log entry.	JXDK, Mongo
Event Code	This is an application event code. In the event that multiple local languages were in use, the event code could be easily tracked across the languages.	[0x899901a1]
Private Tags	This is an optional value that provides additional information.	Process ID (PID) or version number

Log Entry	Description	Example
Message	Message that reflects the application actions and/ or state. This could be a simple string message, JSON or XML object that is compacted to a single line.	Successfully logged into Interplay MAM

Log level descriptions:

Keyword	Value	Description
EMERGENCY	0	System is unusable. Immediate action is required.
ALERT	1	Should be corrected immediately.
CRITICAL	2	Critical conditions. Immediate action is required.
ERROR	3	Error conditions. Cause of the error should be investigated.
WARNING	4	May indicate that an error will occur if action is not taken.
NOTICE	5	Events that are unusual, but not error conditions.
INFO	6	Normal operational messages that require no action.
DEBUG	7	Information useful to developers for debugging the application. This logging level must be manually enabled.
TRACE	8	This is a finer-grain log level that needs to be manually enabled. It can potentially produce very large logs and might introduce performance degradation if left enabled for extended periods.

Viewing the Content of Log Files

From within RHEL

You can search and examine the contents of logs from the Linux command line using the usual Linux tools and commands:

- *less* - Outputs the contents of a file to the screen in a shell; permitting forward and backward movement through the file.

  Press “`q`” to exit the shell.

- *more* - Similar to “less”, but does not allow you to move up and down through the file. “more” is also not presented in a shell. Each page is printed to the screen, providing a percentage of how much of the file has been reviewed. Once the entire file has been displayed, the user is returned to the Linux prompt.

  Press the space-bar to see the next screen or press CTRL-C to exit the `more` command.

- *tail* - By default, this command displays the last ten lines of a log file. Alternatively, you can add a numbered value to specify additional lines:

  ```
  tail <filename>
  ```

  ```
  tail -50 <filename>
  ```

Adding the `-f` switch to the command allows you to view the growing log file in real-time (press CTRL-C to exit the real-time view):

```
tail -f <filename>
```

The same command can be used to simultaneously view multiple log files in real-time. For example the following command displays the last ten lines of both the *edit.log* and *isis.log* in the same shell:

```
tail -f /var/log/avid/edit.log /var/log/avid/isis.log
```

- *grep* - Use the *grep* command to search for regular expressions within a log file from the command line.

  For example the following command searches all log files in the current directory for the term "fail-count":

  ```
  grep fail-count *.log
  ```

  Adding a `-r` option to the same command recursively searches the log files in the current directory and all subdirectories for the specified `<search_term>`:

  ```
  grep -r <search_term> *.log
  ```

- *gzip* - Use the *gzip* command to unzip rotated log files for viewing. Rotated log files are stored as compressed gzip files by default.

  The general form of the *gzip* command for uncompressing *.gz* files is as follows:

  ```
  gzip -d <logfile>.log.gz
  ```

From a Windows System

Logs can be retrieved from the Linux system and reviewed from an external location such as a Windows machine. There are multiple tools that can be used to review the logs. Once such application is called: Notepad++. This free source code editor displays logs through an organized line-item display and enables users to search RHEL logs to quickly find the data they need. Notepad++ can be downloaded from: https://notepad-plus-plus.org/

Retrieving Log Files

Logs can be retrieved from the Linux server through the use of a secure shell (SSH) file transfer protocol (FTP) client — commonly abbreviated SFTP. *WinSCP* (Windows) and *muCommander* (Mac) are free, open-source clients that can securely copy files from a Linux server to a system running Windows or Mac OS. FileZilla, another free open-source utility, can be used in the same way and has the advantage of being available for both Windows and Mac.

WinSCP can be downloaded at the following location: http://winscp.net

muCommander can be downloaded at the following location: http://www.mucommander.com/

FileZilla can be downloaded at the following location: https://filezilla-project.org/

To copy files using WinSCP:

1. Download and install the WinSCP software on a Windows system that has network access to the MCS server.
2. Launch WinSCP.
3. Enter the Host name (or IP address) of your server, User name (*root*), and Password.

The *root* user has the necessary permission levels to establish the connection.

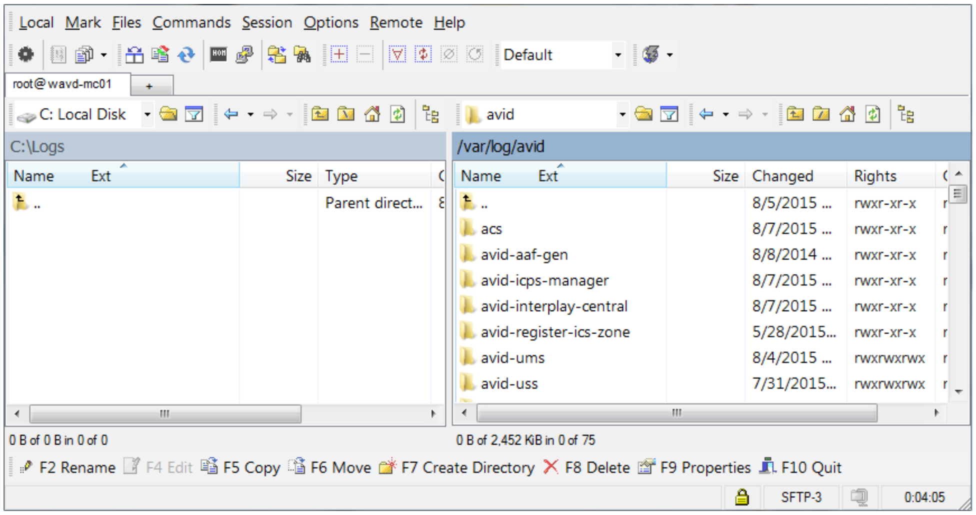
WinSCP uses the standard TCP port 22 for its SSH connection. If you can establish an SSH connection to the server outside of WinSCP, you can use WinSCP.

4. Click Login.

   The following message is displayed: “Continue connecting and add host key to the cache?”

5. Click Yes.

   The WinSCP interface is displayed. The left pane represents your source Windows system. The right pane represents your MCS server.



C:\Logs

Name	Ext	Size	Type
..			Parent direct...

0 B of 0 B in 0 of 0

/var/log/avid

Name	Ext	Size	Changed	Rights
..			8/5/2015 ...	rwxr-xr-x
acs			8/7/2015 ...	rwxr-xr-x
avid-aaf-gen			8/8/2014 ...	rwxr-xr-x
avid-icps-manager			8/7/2015 ...	rwxr-xr-x
avid-interplay-central			8/7/2015 ...	rwxr-xr-x
avid-register-ics-zone			5/28/2015...	rwxr-xr-x
avid-ums			8/4/2015 ...	rwxrwxrwx
avid-uss			7/31/2015...	rwxrwxrwx

0 B of 2,452 KiB in 0 of 75

WinSCP automatically opens in the home directory of the logged in user. Since you logged in as the root user, this is /root on the RHEL machine. This should not be confused with the Linux root directory itself (/).

6. Navigate to the directory on the Windows machine where you want to put log files.
7. Navigate to the directory on the Linux server containing the logs of interest (for example, `/var/log/avid`).
8. Click on the log file of interest to select it or shift-click to select multiple files.
9. Drag and drop the files to the Windows side of the WinSCP interface. Alternately, press the Copy button for more options.

   WinSCP copies the files from the Linux server to the Windows machine.

Altering the Default Logging Levels

Some Avid logs can be altered to increase the amount of detail collected by the log. The default levels should only be adjusted to troubleshoot specific issues and only at the request of Avid Support. Once troubleshooting is complete, the logs should be returned to their default logging levels.

In general this procedure is relevant for all Media Index services as well as messenger and mail services. However the PEBCo service (pam-agent-ctrl), which is a component of systems configured with Media Index for Interplay Production, requires a somewhat different procedure which is also documented below.

To alter the default logging levels for Avid services:

This process applies to all Media Index services as well as messenger and mail services.

1. Identify the service or log that you need information on. This procedure uses the avid-acs-mail service as an example.
2. Using the Linux *vi* editor, open the service configuration file:

   ```
   vi /etc/sysconfig/avid-acs-mail
   ```

3. Add the following line to the file:

   ```
   export ACS_LOG_LEVEL=<log_level>
   ```

   `<log_level>` options are: info, debug, warn, error, trace

4. Save and exit the vi session. Press <ESC> and type: `:wq`
5. To enable the change, the associated service must be restarted:

   ```
   service <service_name> restart
   ```

   Example: `service avid-acs-mail restart`

To alter the default logging levels for the PEBCo services:

1. Sign into the Interplay Administrator on any system that has Interplay Access installed and stop the indexing service in the Production Engine Bus Connector settings.
2. Using the Linux *vi* editor, open the service configuration file:

   ```
   vi /opt/avid/etc/pam-agent-service/logback.xml
   ```

3. This xml file contains several `<appender>` and `<logger>` elements. The log level is configured in the “level” attribute for the logger element. In the following example, the “service.activity.logger” element is configured for the default logging level of INFO:

   ```
   <logger name="service.activity.logger" level="INFO"
       additivity="false">
       <appender-ref ref="SERVICE_ACTIVITY_APPENDER" />
   </logger>
   ```

   The logging level for the following elements can be adjusted from INFO to DEBUG:

   - engine_activity.log - Lists all changes detected on the Interplay Engine (input): `<logger name="engine.activity.logger"`
   - service_activity.log - Lists all changes sent out to the index (output): `<logger name="service.activity.logger"`
   - pamagent.log - Internal state of the PEBCo service. Error messages are often found in this log: `<logger name="com.avid.ime.pam"`
   - jxdk.log - Contains log and error messages from the PAM Engine connection library: `<logger name="net.nxn.JXDK"`

4. Save any changes and exit the vi session. Press <ESC> and type: `:wq`
5. Verify that the PEBCo instance name through the `pam-agent-ctrl` script:

   ```
   pam-agent-ctrl list
   ```

   The screen displays a list of all available PEBCo (pam-agent) instances on this machine. The following is a sample output from a single server:

   ```
   [root@wavd-mcs01 ~]# pam-agent-ctrl list
   wavd-doc01
   ```

6. Restart the PEBCo service on the MediaCentral server:
   - For single-server configurations, type the following command:

     pam-agent-ctrl start <*instance_name*>
   - For cluster configurations, type the following command on one of the nodes in your cluster:

     crm resource start *AvidPamAgent-<instance_name>*
7. Once the service has been restarted, use the Interplay Administrator to start the service in the Production Engine Bus Connector settings.

Only alter the logging level if instructed to do so by Avid Support. If you have altered the default level, return the file to its original state once troubleshooting is complete. As a reminder, be sure to restart the affected service after altering the logging levels.

Important Log Files at a Glance

The following tables detail the name, location and purpose of the logs found on an MCS server.

RHEL Logs in /var/log

The following table presents the standard RHEL logs found in the /var/log directory:

Log File	Description
/var/log/anaconda.log	Linux installation messages.
/var/log/boot.log	Information pertaining to boot time.
/var/log/btmp.log	Failed login attempts.
/var/log/cron	Information logged by the Linux cron daemon.
/var/log/dmesg	Information about hardware detected by the kernel at boot time. The Linux *dmesg* command shows the contents of this log.
/var/log/dracut.log	Log file of the Linux *initramfs* image creation process.
/var/log/lastlog	Most recent log-in for all system users. Use Linux *lastlog* command to view the contents of this log.
/var/log/maillog	Mail server log.
/var/log/mcelog	The *machine check events* (memory and CPU error) log.
/var/log/messages	Global system messages, including startup messages, logins, packet logging.
/var/log/secure	Authentication and authorization messages.
/var/log/spooler	Usenet and uucp log.
/var/log/tallylog	Failed login attempts.
/var/log/wtmp	Current login records. Use the Linux *who* command to display the contents.
/var/log/yum.log	Information about packages installed using Linux *yum* utility.

RHEL Subdirectories in /var/log

The following table presents the standard RHEL subdirectories found in the /var/log directory:

Log File	Description
/var/log/audit	Logs stored by the RHEL audit daemon.
/var/log/ConsoleKit	Logs stored related to user sessions. Deprecated.
/var/log/cups	Logs related to printing.
/var/log/httpd	The Apache web server access and error logs. As of ICS 1.8 Apache is no longer used.

/var/log/ntpstats	Logs relating to the NTP daemon.
	To enable NTP logging, add lines similar to the following to /etc/ntp.conf:
	`statistics clockstats cryptostats loopstats peerstats`
	`logconfig =all`
	`logfile /var/log/ntp`
	`statsdir /var/log/ntpstats/`

/var/log/prelink	Information related to the Linux *prelink* program that speeds up the startup process.
/var/log/rhsm	Logs related to the Red Hat Subscription Manager.
/var/log/sa	Information collected and stored by the Linux *sar* performance monitoring utility (CPU, memory, I/O, network statistics, and so on). The *sar* utility is part of the larger Linux *sysstat* package. It reports local information only (i.e. it is not cluster-ready).
/var/log/samba	Logs related to the Samba programs.
/var/log/sssd	Information stored by the Linux *system security services daemon* responsible for access to remote directories and authentication.

Postgres Logs

The PostgreSQL database is used for user management and attributes data. Its log files are stored in the following locations:

Log File	Description
/var/lib/pgsql/9.1/data/pg_log/	Location of the logs for a single-server configuration.
/mnt/drbd/postgres_data/pg_log/	Location of the logs on the master node of a cluster.

Avid Logs in /var/log

The following table presents logs specifically related to MCS and related systems found in /var/log and its associated subdirectories:

Log File	Description
/var/log	• **MediaCentral_Services_<version>Build<number>_Linux.log** - Logs any errors encountered during the an MCS software installation. • **ICS_installer_<version>_<build>.log** - Similar to the log file above, but related to legacy "Interplay Central Services" installations. • **ICS_install.log** - Similar to the log file above, but related to legacy "Interplay Central Services" installations. • **fuse_avidfos.log** - Logs related to the Linux fuse interface, used by the *avid-isis* back-end service to mount Avid shared storage workspaces. • **pacemaker.log** - Information related to the Cluster Resource Manager. This log file is only available in clustered MCS configurations.
/var/log/avid	• **avid-db.log** - Log file of the avid-db database management tool. All of the following ICPS / MCPS (playback service) logs are overseen by the avid-all service: • **config.log** - MCS UX configuration information, as found in the System Settings panels. Produced by *avid-config* service. • **edit.log** - Logs related to the back-end systems, including host and log-in information, timeline warnings, and so on. Helpful when troubleshooting Avid ISIS, Avid NEXIS, and Interplay Production login issues. Produced by *avid-edit* service. • **fps.log** - Flash Player Security (FPS) information, relating to the player appearing in MCS UX. Produced by *avid-fps* service. • **isis.log** - Information pertaining to Avid ISIS and Avid NEXIS mounts and connections. Produced by *avid-isis* service. • **jips.log** - Java Interplay Production service. Contains information pertaining to low-level connections between the MCS back-end services and the Interplay Production services used to obtain AAF metadata. Produced by *avid-jips* service. • **monitor.log** - Log file for the avid-monitor service in cluster configurations. • **reconfigure.log** - Activity associated with running "*service avid-all reconfigure*", which runs during setup. • **spooler.log** - Information relating to playback. Produced by *avid-spooler* service.

Log File	Description
/var/log/avid/acs	• **avid-acs-attributes.log** - Log file for the avid-acs-attributes service which stores service configuration attributes. • **avid-acs-federation.log** - Log file for the avid-acs-federation service which stores bus configuration information for multi-zone. • **avid-acs-infrastructure.log** - Log file for the avid-acs-infrastructure service which is used to track bus server connection information used by the Bus Access Layer component. • **avid-acs-mail.log** - Log file for the avid-acs-mail service. • **avid-acs-messenger.log** - Log file for the avid-acs-messenger service. • **avid-acs-monitor.log** - Log file for the avid-acs-monitor service which logs information about the app used to view Service Status for a system. • **avid-acs-registry.log** - Log file for the avid-acs-registry service which manages a registry of service instances that are present on the Bus. • **avid-acs-service-manager.err** - Log file for avid-acs-service-manager. • **avid-acs-service-manager.log** - Log file for avid-acs-service-manager. • **busaccess_cpp.log** - Log file for C++ Bus Access Layer.
/var/log/avid/acs/gateway/	avid-acs-gateway.log - Log file for the avid-acs-gateway service found in MediaCentral Platform Services v2.8.x and later.
/var/log/avid/acs/acs-query	Directory for logs generated by acs-query tool. Log files will named: acs-query.<user name>.<date>.log
/var/log/avid/acs/wrapper/	wrapper-avid-acs-gateway.log - Log file for the avid-acs-gateway service found in MediaCentral Platform Services v2.7.x and earlier.

/var/log/avid/ansible/	Contains multiple log files created by the scripts used to create the sharded Mongo configuration in MediaCentral Platform Services v2.6.x and later. Each script creates its own log file and each includes a time/date stamp. For example:
	`mongo-create-configuration_2016-09-08_18:48.log`
	`mongo-create-configuration_2016-09-13_15:09.log`
	`mongo-playbook-setup_2016-09-07_20:31.log`
	`mongo-playbook-setup_2016-09-07_20:35.log`
	`mongo-playbook-clean-all_2016-09-13_15:19.log`
	`mongo-playbook-remove-arbiter_2016-09-07_20:59.log`
	`mongo-playbook-remove-arbiter_2016-09-08_19:15.log`

/var/log/avid/avid-aaf-gen/	AAF Generator logs. This is the service responsible for saving sequences. There are generally five instances of this service running on an MCS server. Each service creates logs in its own sub-folder (log_1, log_2, etc).
/var/log/avid/avid-asset/	Contains logs related to the avid-asset service introduced in MCS v2.9. Useful for troubleshooting issues with the “Mixed Sequence editing” feature.
/var/log/avid/avid-asset-gc/	Contains logs related to the avid-asset-gc service introduced in MCS v2.9. Useful for troubleshooting issues with the “Mixed Sequence editing” feature.

Log File	Description

/var/log/avid/avid-ccc	Logs related to the Closed Captioning Service (if installed).
	If more detailed information is required, native (C++) debug logging can be enabled by adding the following two lines to the CCS configuration file at `/etc/sysconfig/avid-ccc`:
	`export KEEP_TEMP_FILES=1`
	`export NATIVE_DEBUG_LOGGING=1`
	Restart the avid-ccc service to enable the change.
	• On a single server: service avid-ccc restart
	• In a cluster: crm resource restart AvidCCC
	Once the desired data has been logged and captured, remove the two lines from the configuration file and restart the service again.

/var/log/avid/avid-iam	Logs related to the Identity and Access Management service (avid-iam). These logs can be helpful when troubleshooting the sharded Mongo configuration in MCS v2.6 and later.
/var/log/avid/avid-icps-manager	The icps-manager is a web service that relays data between the flash player and the MCS player services. In MCS v2.5.2 and later, the logging level for the avid-icps-manager service can be altered to provide additional detail if necessary. To alter the logging, change the “log-level” parameter of the service config file at: `/opt/avid/avid-icps-manager/icps_mgr_config.yaml`. The default log-level is “info”. This can be changed to “debug” for more detail. Restart the avid-icps-manager service to enable the change. *Only alter the logging if instructed to do so by Avid Support. Return the logging to “info” and restart the service once troubleshooting is complete.*
/var/log/avid/avid-interplay-central	• ***YYYY_MM_DD***.request.log - Daily request logs • **acs-bal-*YYYY-MM-DD*.0.log** - • **interplay_central_#.log** - MediaCentral server log. Helpful for troubleshooting a variety of problems including login issues and failed searches. • **osgi.log** • **osgi-framework.log** • **service_startup.log** • **uls.log** This directory also contains the following sub-folders: • client - Contains log messages pertaining to the client application • health check - Health monitoring logs • icps • interplay - Contains Interplay production connection logs • performance - Contains Interplay performance logs. This logging is disabled by default.

Log File	Description
/var/log/avid/avid-register-ics-zone	*Reserved for future use.*
/var/log/avid/avid-ums	• **audit.log** - Added with MCS v2.5, this log provides information about changes in roles, users, and groups. Events are logged with date, time, message (such as “User has been created”) and other information. • **avid-iam.log** - Added with MCS v2.5, contains events related to communication between avid-ums and avid-iam • **bus-errors.log** - Contains errors from bus-logs.log • **bus-logs.log** - Contains events related to communication between avid-ums and java bal • **error.log** - Contains only ERROR events from avid-ums service • **importer.log** - Contains LDAP user import events • **service.log** - General log for the User Management Service • **session.log** - User session information. Contains information on what user logged in at what time. It also logs the IP address used to make the connection. • **statistics.log** - Contains events from StatsD client in avid-ums *Log files are created during the avid-ums service startup. To recreate all user management logs, restart the avid-ums service.*
/var/log/avid/avid-upstream	Logs related to the Avid Upstream service.
/var/log/avid/avid-uss	Logs related to the User Setting Service.
/var/log/avid/media-index	Logs related to Media Index. These logs are only available if Media Index has been configured. • **avid-acs-autocomplete.log** - Autocomplete service log file • **avid-acs-media-index-configuration.log** - Configuration service log file • **avid-acs-media-index-feed.log** - Feed service log file • **avid-acs-media-index-permission.log** - Permissions service log file • **avid-acs-media-index-status-provider.log** - Status provider log file • **avid-acs-media-index-thesaurus.log** - Thesaurus service log file • **avid-acs-search.log** - Search service log file • **avid-acs-search-query.log** - Log all search queries (disabled by default) To enable this log, add the “ACS_ENABLE_QUERY_LOGGER” variable to the `avid-acs-search` configuration file at: `/etc/sysonfig/` Configuration information for the logs above is located at: `/etc/sysconfig/` Log rotation information for the logs above is located at: `/etc/logrotate.d/` Additional logs related to Media Index: • **avid-acs-search-import-compatibility-layer.log** - Import service v0 log file • **avid-acs-search-import-failed-assets-compatibility-layer.log** - Import service v0 failed assets log file

Log File	Description
/var/log/avid/media-index (*continued*)	By default, the configuration information for these files is located at: `/opt/avid/lib/com.avid.search.import/compatibility.layer/libs/logconfig/logback-compatibility.xml` However, this path can be changed by altering the following variable: IMPORT_COMPATIBILITY_LOG_CONFIG_PATH Log rotation information for the logs above is located at: `/etc/logrotate.d/` Additional logs related to Media Index: • **avid-acs-search-import.log** - Import service v1 log file This service allows you to change the logging level through the Avid ACS Monitor Tool. This enables users to alter the logging level without restating the service. • **avid-acs-search-import-failed-assets.log** - Import service v1 failed assets log file. • **avid-acs-search-import-init-requests.log** - Import service v1 init log file By default, the configuration information for these files is located at: `/opt/avid/lib/com.avid.search.import/service/libs/logconfig/logback.xml` However, this path can be changed by altering the following variable: IMPORT_LOG_CONFIG_PATH Log rotation information for these logs is located at: `/etc/logrotate.d/`
/var/log/avid/pam-agent-service/	This folder contains sub-folders for each configured Pam Agent instance with corresponding log files for that instance. These logs are only available for systems configured with Media Index.
/var/log/avid/qm	Quality Manager (relink) logs.
/var/log/avid-syslog	• **edit.log** - deprecated • **spooler.log** - deprecated
/var/log/cluster	Corosync log files. These log files are only available in clustered MCS configurations.
/var/log/elasticsearch	Logs related to the elasticsearch component of Media Index. Logs are only available if Media Index has been configured. • ***<ClusterName>*.log** - Where <*ClusterName*> is the host name of the single node or virtual cluster name of the MCS system. • ***<ClusterName>*_index_search_slowlog.log** - Where <*ClusterName*> is the host name of the single node or virtual cluster name of the MCS system. This file logs data for systems that are returning search queries at a slower than normal rate. • ***<ClusterName>*_index_indexing_slowlog.log** - Where <*ClusterName*> is the host name of the single node or virtual cluster name of the MCS system. This file logs data for systems that are indexing data at a slower than normal rate. Configuration and rotation information for these logs is found at: `/etc/elasticsearch/logging.yml`

Log File	Description
/var/log/elasticsearch-tribe	Logs related to the elasticsearch component of Media Index. These logs are only available if Media Index has been configured. • **elasticsearch.log** - Log for the elasticsearch-tribe service. • **elasticsearch_index_search_slowlog.log** - This file logs information for tribe queries that are running at a slower than normal rate. • **elasticsearch_index_indexing_slowlog.log** - This file logs information for tribe indexes that are running at a slower than normal rate. Configuration and rotation information for these logs is found at: `/etc/elasticsearch-tribe/logging.yml`
/var/log/glusterfs	Logs for the GlusterFS file replication software. These log files are only available in clustered MCS configurations.
/var/log/mongodb	Log files for MongoDB, including services for sharded MongoDB.
/var/log/nginx	• **access.log** - Log file for the nginx service. • **error.log** - Log file for the nginx service.
/var/log/rabbitmq	RabbitMQ log files.

Media Distribute Logs

The following table presents log information specific to Media Distribute. Media Distribute is a separate install package which will not be found on all systems.

Log File	Description
/var/lib/apche-servicemix/data/logs/servicemix	No description
/usr/share/apache-servicemix/data/log/ • servicemix.log • wrapper.log	If an error occurs when validating a Distribute profile in the MCUX System Settings, more information about the failure might be found in these logs.

MediaCentral Distribution Service Logs

The following table presents log information for the MediaCentral Distribution Service (MCDS); supported by Interplay Production send-to-playback workflows. MCDS is generally installed on a Windows server hosting other Interplay Production services.

Log File	Description
C:\ProgramData\Avid\Interplay Central Distribution Service or C:\ProgramData\Avid\MediaCentral Distribution Service	• STPService_nn.log - Messages from the MediaCentral Distribution Service • STPTimerTask_nn.log - Messages for the job status automatic clean-up
/var/log/avid/avid-interplay-central/interplay_central_0.log	Located on the MCS server, this log contains the information related to MCDS.

Browser Logs

The following table presents log information for the web browsers supported by MediaCentral UX.

Log File	Description
Chrome	Select “More tools” from the Chrome menu and select “JavaScript console”
Safari	Safari crash logs: /Applications/Safari.app/Contents/MacOS/Safari

MediaCentral | UX Connector for Adobe Premiere Pro CC Logs

The following table lists the logs found on Mac and Windows clients for the Connector.

Log File	Description
Windows	%temp%\com.avid.central.adobe.log
Mac	$TMPDIR/com.avid.central.adobe.log

Mobile Device Logs

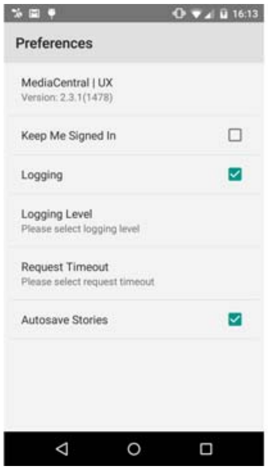
Logs are available for both iOS and Android devices. Logs can be sent directly from the device through e-mail or can be obtained by connecting the device to a host system.

Enabling and Emailing Device Logs

Logging is not enabled by default and must be manually selected per device. To ensure best performance of the device, logging should only be enabled temporarily to create a log for a specific issue.

To enable logging for iOS and Android devices:

1. Sign in to your mobile client.
2. Select the application menu to access the Preferences or Settings.
3. Select the option to enable logging. In the example below, the Android app is pictured on the left and the iOS app is pictured on the right.



16:13

Preferences

MediaCentral | UX
Version: 2.3.1(1478)

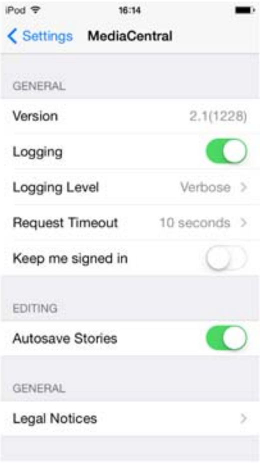
Keep Me Signed In □

Logging ☑

Logging Level
Please select logging level

Request Timeout
Please select request timeout

Autosave Stories ☑



iPod 16:14

< Settings MediaCentral

GENERAL

Version 2.1(1228)

Logging ●

Logging Level Verbose >

Request Timeout 10 seconds >

Keep me signed in ○

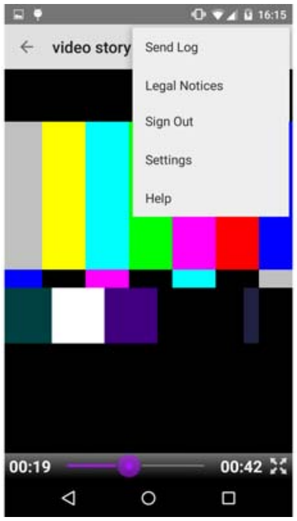
EDITING

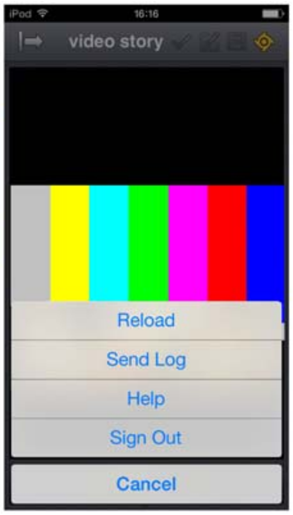
Autosave Stories ●

GENERAL

Legal Notices >

4. If directed by Avid support, adjust the Logging Level:
   - Verbose
   - Info
   - Warn
   - Error
5. Perform any operations related to the issue you would like to reproduce.
6. Once you have reproduced the issue, select “Send Log” from the application menu. In the example below, the Android app is pictured on the left and the iOS app is pictured on the right.





7. Send an e-mail with the log to yourself or an Avid representative for analysis.

Obtaining Device Logs Through a Host

If multiple log files exist, collecting and sending all logs in a single e-mail might be more desirable than sending them through the device itself. iOS devices allow the user to connect the device to a Mac computer and browse to the location of the log files.

Android devices often only present the SD card storage to the user and not the internal storage where the application and logs are located. The following process is not applicable to Android devices.

To collect logs through a Mac host:

1. Connect your device to a Macintosh computer
2. Open iTunes and navigate to Device > Apps.
3. In the Apps list, select MediaCentral UX.
4. In the MediaCentral UX Documents list, select the Logs folder.
5. Click “Save to” or drag the folder to a location on your computer.
6. Zip the folder and send it to Avid as an e-mail attachment.

Avid
75 Network Drive
Burlington, MA 01803-2756 USA

Technical Support (USA)
Visit the Online Support Center at www.avid.com/support

Product Information
For company and product information, visit us on the web at www.avid.com