**COMMON LATTUS ISSUES**

These issues were discovered at Apple MVP, but we see these on other Lattus systems. Some issues can be due to bugs, but do not assume anything--check them out.

Contents

[Lattus HOW TO change the default root password 3](#_Toc476037031)

[Failed to save job warnings in EVENT LOGS 4](#_Toc476037032)

[Event Message "Failed to {retrieve NICs ; retrieve application guids ; retrieve disk guids} from model" 4](#_Toc476037033)

[Too many instances running for policy monitoring\_storagepool, skipping execution what to do 5](#_Toc476037034)

[4TB disks for the Lattus S20 nodes are different for the older Aeaon Nodes as opposed to the newer Sanmina Model 2 S20 nodes 6](#_Toc476037035)

[SWAP USAGE Swap usage is over 90% 7](#_Toc476037036)

[Lagging keys 8](#_Toc476037037)

[MetaStore is DOWN 9](#_Toc476037038)

[Maintenance agent has status halted 9](#_Toc476037039)

[Machine agent is down 11](#_Toc476037040)

[More than 50 tlogs More than 30 tlogs 11](#_Toc476037041)

[ARA005 - How to reduce the number of tlf files in the Arakoon tlog directory 11](#_Toc476037042)

[MetaStore framework is lagging keys on machine 16](#_Toc476037043)

[DRIVE IS READ ONLY 17](#_Toc476037044)

[The first step after a disk goes degraded 21](#_Toc476037045)

[Disk not detected 23](#_Toc476037046)

[Kernel dmesg errors 23](#_Toc476037047)

[EXAMPLE OF DISK FAILURES PENDING VIA Kernel dmesg errors 27](#_Toc476037048)

[Machine was rebooted 28](#_Toc476037049)

[Disk failures for 28](#_Toc476037050)

[Blockstore temporarily offline 32](#_Toc476037051)

[Low disk safeties 33](#_Toc476037052)

[[storageXXX] needs to be diagnosed and replaced 35](#_Toc476037053)

[BDY015 - How to manually decommission a disk. 35](#_Toc476037054)

[STORAGExxx is HALTED 36](#_Toc476037055)

[SCS030 - HowTo Configure An Automated Reboot Of A Hung/Frozen node 36](#_Toc476037056)

[Filesystem errors on disk 37](#_Toc476037057)

[Failure while executing event handling logic 37](#_Toc476037058)

[Failed to execute monitoring rule 38](#_Toc476037059)

[ALL ASSOCIATED WITH MEMORY MANAGEMENT BUG (QVC memory cleanup bug) 39](#_Toc476037060)

[BOTH OF THESE WERE RELATED TO DISK REPLACEMENT 41](#_Toc476037061)

[Machine was rebooted 41](#_Toc476037062)

[New empty disk(s) detected 41](#_Toc476037063)

[Error retrieving repair statistics for daemon with id 41](#_Toc476037064)

[Errors detected on machine Error retrieving block store information 41](#_Toc476037065)

[Error retrieving storage daemon black lists 42](#_Toc476037066)

[Storage pool is more than 70% full 43](#_Toc476037067)

[Unverified objects 44](#_Toc476037068)

[ZUFFA SR3582806 FYI storage011 issues 45](#_Toc476037069)

[UPGRADE ISSUE 47](#_Toc476037070)

[Error retrieving storage daemon info for daemon with id 48](#_Toc476037071)

# Lattus HOW TO change the default root password

**Procedure**

To change the default root password:

1. Open an SSH connection to the Lattus system (Management Controller Node).
2. Enter qshell by typing the following command and pressing **Enter:**
   * 1. **qshell**
3. Set a variable to the http database api connection to the cloud:
   * 1. In [1]: **api=i.config.cloudApiConnection.find('main')**
4. Use the appropriate command to set a variable that contains the list of machine (node) guids:

* To change the root password on *all nodes (Controller Nodes and Storage Nodes):* In [2]: **machinelist = api.machine.find()['result']**

* To change the root password on *all Controller Nodes*:In [2]: **machinelist = api.machine.find(machinerole='CPUNODE')['result']**

* To change the root password on *all Storage Nodes:* In [2]: **machinelist = api.machine.find(machinerole='STORAGENODE')['result']**

* To change the root password on *a single node (Controller Node or Storage Node):* In [2]: **machinelist = api.machine.find(name='Storage8')['result']**

1. Change the password on each node in the previously set variable ‘machinelist’.

In [3]: **for machine in machinelist:**

**...: api.machine.changePassword(machine, 'root', ‘<new-password>')**

...:

...:

Out[3]: {'jobguid': '7fb94e33-fba0-495f-b7ee-8276fb7434b2', 'result': True}

Out[3]: {'jobguid': '94226085-e4c1-4b0b-97ef-20c0f67e1ef9', 'result': True}

This password change is dynamic and happens almost immediately on all applicable nodes. You do not need to do anything on the individual nodes.

# Failed to save job warnings in EVENT LOGS

Failed to save job amplistor.save\_monitoring\_data.amplistor\_save\_monitoring\_data

Failed to save job amplistor.save\_monitoring\_data

Failed to save job amplistor.aggregateStoragepoolInfo

[**Bug 57282**](http://bo/bugzilla/show_bug.cgi?id=57282)

The controller logs did not help pin this down.

Can you run a healthcheck on this lattus and send me the output from that I think it will let me know which node is causing this.

**qshell -c "q.amplistor.healthCheck()" | tee healthCheck.txt**

**And also please send this log it is not included in the alllogs**

/opt/qbase3/apps/log\_collector/log\_collector\_trigger.py -t alldaemons -C -d 2017-02-02

Can try fixing with this:

qshell

q.manage.applicationserver.checkStatus()

q.manage.applicationserver.restart()

exit()

# Event Message "Failed to {retrieve NICs ; retrieve application guids ; retrieve disk guids} from model"

Comment: All of these calls require access to the cloud\_api, which could have been temporarily unavailable due to log rotation which sends a SIGHUP to apache. These errors can be ignored **if** they match the corresponding messages for an Apache restart, in the 'system messages' file.

**Example:**

den-storage001 (54:A0:50:87:11:CDErrors detected on machine den-storage001: Failed to retrieve disk guids from model

**Example**

Fri Feb 22 23:30:53 2013] [notice] SIGHUP received. Attempting to restart  
[Fri Feb 22 23:30:53 2013] [notice] mod\_python: Creating 64 session mutexes based on 256 max processes and 0 max threads.  
[Fri Feb 22 23:30:53 2013] [notice] mod\_python: using mutex\_directory /tmp  
[Fri Feb 22 23:30:53 2013] [notice] Apache/2.2.22 (Unix) mod\_ssl/2.2.22 OpenSSL/0.9.8g mod\_python/3.3.1 Python/2.6.2 configured – resuming normal operations

# Too many instances running for policy monitoring\_storagepool, skipping execution what to do

- per HGST 1135:  Too many instances running for policy monitoring\_storagepool, skipping execution

Per SR3572904: “We deleted the running policy job and rebooted ¿ that cleared the message.” - Phil Thorsten

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

This condition can occur when a policy job is stalled and fails to complete before the timeout for another job is scheduled to start.

In the case of the 'Aggregate Storagepool Info' policy job, this is scheduled to run every 30 minutes to update the state files that report storage capacity/usage etc.

If you look in the CMC, under 'Dashboard' -> 'Administration' -> 'Lattus Management' -> 'Logging' -> 'Policies' you will see a list of completed (or still running policy jobs. You should see a job titled 'Aggregate Storagepool Info' that is > 30 minutes old and in the running state.

**This condition can be cleared by restarting the workflowengine on the management controller (cpunode1 based on the logs uploaded). You can do this via qshell:**

q.manage.workflowengine.restart()

root@controller01:~# **qshell**

Welcome to qshell

? -> Introduction to features.

help() -> python help system.

object? -> Details about 'object'.

object?? -> Extended details about 'object'.

Type q. and press [TAB] to list qshell library

Type i. and press [TAB] to list interactive commands

In [1]: **q.manage.workflowengine.restart()**

Stopping the workflowengine.

Stopped the workflowengine.

Starting the workflowengine.

Waiting for initialization

Workflowengine started

In [2]: **quit()**

Do you really want to exit ([y]/n)? y

root@controller01:~#

# 4TB disks for the Lattus S20 nodes are different for the older Aeaon Nodes as opposed to the newer Sanmina Model 2 S20 nodes

The question is how do you tell the difference with the nodes?

Probably not surprising, but the logs (at least as far as I know) will not tell you which type of node is installed. For example, Apple has Sanmina Model 2 nodes and Crown Media has Aeaon nodes.

There are a couple of ways to confirm the difference:

1) Run the following command after logging in with ssh to the Storage Nodes in question (IP can be obtained from the Lattus GUI under Hardware > Storage Nodes, then login form an existing CMC ssh session):

(Phil actually shared this command a year or so ago, when we were trying to figure out the node type at Apple)

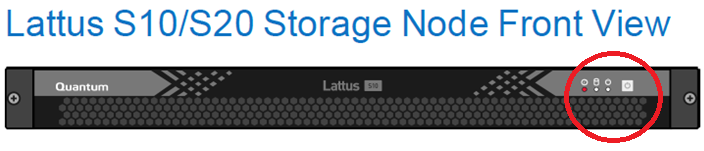
# dmidecode | grep Vendor

Aeaon:                 Vendor: American Megatrends Inc.

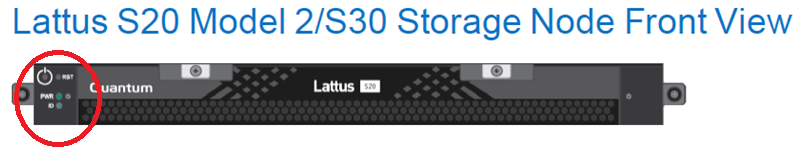
Sanmina:             Vendor: Newisys

2) The customer can physically check the Power LEDs are located in different positions. **On an Aeaon they are on the right (looking directly at the front) and Sanmina are on the le**ft.

AEAON



SANMINA



# SWAP USAGE Swap usage is over 90%

Swap usage is over 90% from Quantum-1276 SR3651166

Linux divides its physical RAM (random access memory) into chunks of memory called pages. Swapping is the process whereby a page of memory is copied to the preconfigured space on the hard disk, called swap space, to free up that page of memory.

When physical memory is used upto 90% then it will start using the swap space to decrease the memory pressure from physical RAM.

**Swappiness**

The swappiness sysctl parameter represents the kernel's preference (or avoidance) of swap space. Swappiness can have a value between 0 and 100, the default value is 60. Setting this parameter to a low value will reduce swapping from RAM, and is known to improve responsiveness on many systems.

Swapping is currently set to 90 in your controller.

Please verify the value using following command:

# **cat /etc/sysctl.conf | grep swappiness**

Monitoring agent on each node will check the swap uses and if swap usage is more than the 'swappiness' assign value than it will generate an event.

Workaround as below:

**swapoff -a**

**swapon -a**

This will clear the event.

Second option to remove the check\_swap\_usage from monitoring agent that way the event won't generated.

It is your call.

Please issue following command to see which process are using more memory.

# **for file in /proc/\*/status ; do awk '/VmSwap|Name/{printf $2 " " $3}END{ print ""}' $file; done | sort -k 2 -n -r | less**

Issue the above command when node has an event about swap usage.

# Lagging keys

Arakoon – Lagging keys

Abstract

Arakoon is consistent, distributed database. It serves as the key database for several components of the Amplistor storage solution. In the distributed database each node act as a state machine. Some times a node of the cluster will run a few states behind. This document describes the general mechanism behind using paxos to implement a database as a distributed state machine. It explains in which situation some of the nodes might run a few states behind and when it is a problem.

Arakoon – A distributed key value store

An arakoon cluster consists of three nodes that are on different machines. Each node has a database. The nodes use the paxos protocol to achieve full consistency. Each node is a state machine. When a cluster is stable, one node is in the master state and the two other nodes are in the slave state. For consistent requests, the client should always send requests to the node which is in master state. This node is the only node who is allowed to answer or process requests. The protocol between the client and server ensures easy discovery of the current master.

When the nodes are start, they immediately negotiate who becomes the master. When a node has become a master, it will always try to stay master. If the master can not stay master (for example due to connection problems), the slaves will elect a new master. This ensures the cluster is always available.

Before applying updates to the database, the master sends this request to the two slave nodes. When one of the two slaves has accepted this update and has send an accepted message back to the master, the update gets applied to the database of the master. When the slaves get the message from the master that the update was applied, they also apply the update to their databases,

When the master receives an update from the client, the master assigns a transaction id to it. This transaction id is an increasing counter. The nodes keep track of the transaction id of the last update that was applied to the database. The current state of the database is as such the application of all state transitions to the database.

Running behind and catching up

If at a moment a slave notices that the transaction id of its last update is lower than the transaction id of the last update of the master, it will start a catchup from the master. This means that it will ask the master to send all the updates the slave is missing over the network. When this catchup procedure ended successfully, the slave is again up to date.

To facilitate this catchup process, each node also keeps a transaction log of updates that are applied to the database. In case loss of the primary database, a node can always recover from the local or remote transaction logs. If we would never cleanup the transaction logs, we would end up with a huge amount of them. The recovery process would take a very long time to rebuild the database applying all

transaction logs. This is why we need the *collapse transaction logs* procedure: it takes a backup of the hot database, stores this backup on the HDD and then removes all transaction logs that have become redundant (a transaction log is redundant if it only contains updates that are in the backupped database)

This c*ollapse transaction logs* procedure is run daily, but it is never run on nodes of the same cluster simultaneously. When the master nodes need to run the c*ollapse transaction logs* procedure, it will ask the two slaves to elect a new master. The master can then become a slave and run the procedure.

Lagging keys

A warning like *"Node node\_1\_9005 on MetaStore stornextdata01 is lagging keys on machine"* means that the mentioned slave node is more than 300 updates behind on the master and the other slave. This can have **several root causes:**

**1. There are or have been connection problems, this slave could not connect to the master and therefore did not receive any updates.**

**2. The process or the entire machine is/has been down.**

**3. The slave process is running maintenance tasks such as the c*ollapse transaction logs* procedure. This causes the process to become slower in accepting updates from the master. As long as the other slave is not doing maintenance, this will not influence the speed for the client.**

In any of these three cases, the lagging node should start a catchup from the master. So this warning should solve itself.

**If it is still recurring after a couple of hours**, this might indicate that there is really a problem with this node. In this case it is severe.

A node that does not catchup anymore will not be able to join the master election. If another node of the cluster goes down, the cluster will become unavailable. Leading to unavailability of the data service (both for reads and writes).

This means that in case the event does not go away, it is important to escalate to support.

# MetaStore is DOWN

arakoon attempts to recover from this situation automatically including doing a master failover

if you get logs get them from all controllers

Quantum-803. The monitor agent raises the event in the short window after the arakoon master is dropped and a new one is elected during the tlog collapse workflow. The situation is self correcting and can be considered benign at this point.

# Maintenance agent has status halted

Subject: Maintenance agent '76902987' has status halted on environment 'CX1421CKC00014'  
An event occurred:  
Maintenance agent '76902987' has status halted  
Possible solution:  
Validate why the application isn't responding anymore (applicationserver/port down, network issues...).  
Severity: ERROR  
Machine: marcom-nwk-lt-st130  
Machine type: STORAGENODE

From st130 syslog alllogs

This node keeps rebooting

Line 798: Nov 5 12:15:12 localhost kernel: imklog 5.8.6, log source = /proc/kmsg started.

Line 2051: Nov 5 12:55:20 localhost kernel: imklog 5.8.6, log source = /proc/kmsg started.

Line 3360: Nov 5 14:28:35 localhost kernel: imklog 5.8.6, log source = /proc/kmsg started.

Line 4790: Nov 5 17:53:50 localhost kernel: imklog 5.8.6, log source = /proc/kmsg started.

Line 6131: Nov 5 20:09:00 localhost kernel: imklog 5.8.6, log source = /proc/kmsg started.

Line 7455:

Line 9428: Nov 6 10:01:06 localhost kernel: imklog 5.8.6, log source = /proc/kmsg started.

Line 10646: Nov 6 10:14:09 localhost kernel: imklog 5.8.6, log source = /proc/kmsg started.

Line 11863: Nov 6 10:23:55 localhost kernel: imklog 5.8.6, log source = /proc/kmsg started.

   We saw this issue a while back, and the fix was to upgrade the BIOS on the storage nodes. It may be something similar, or there could be some communications issues between this node and the controller nodes.

This was the Bug Matt found

Maintenance agent has status halted

QUANTUM-1062 Reduced **maintenance** **agent** threads from 4 to 2; reduced **maintenance** **agents** from 2 to 1 SR3543180

[**Bug 58802**](http://bo.quantum.com/bugzilla/show_bug.cgi?id=58802) - Maintenance agent <agent\_number> has status halted

too many maintenance agents and parallel tasks used up the memory on the

storage node. There is a patch coming to correct garbage collect on the memoryand prevent this from occurring.The work around for this ticket was to reduce the maintenance agents from 2 to

1 and also to reduce the parallel tasks from 4 to 1.

After the work around the maintenance agent no longer halted.

DSS-1990, memory patch for S30 memory allocation during repair process.

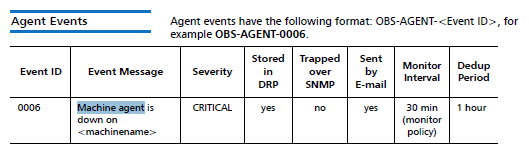
Raising triage value to get on to the 3.8 request list

Expectation is the memory allocation patch will be in the 3.8 list.

# Machine agent is down

**workflowengine HanController01** (44:A8:42:1B:6E:71) CRITICAL 2 2015-10-01 16:17:49 2015-10-01 16:48:04

Machine agent is down on HanStorage002



# More than 50 tlogs More than 30 tlogs

**LATTUS-M/ More than 50 tlogs found on node node\_5\_9003 of MetaStore data03 on machine [controller6]**

This is an existing bug. Exposure if crash, then Arakoon recovery would require another instance.

More than 50 tlogs found on node - this gets raised when the number of transaction logs exceeds the threshold you can configure in the transaction log collapsing policy. This situation should correct itself the next time the collapsing policy runs. Right now there is a problem changing this policy that Amplidata is working on. The reason we want the tlogs to be kept down, is to limit the time necessary to reconstruct a database, the amount of .tlf files (compressed metastore tlog files), if that ever becomes necessary.

SEE **Bug 56542** QUANTUM-1084

Lattus Tracker: Tlog collapsing policy not able to keep up with default settings.

# ARA005 - How to reduce the number of tlf files in the Arakoon tlog directory

Do not execute this procedure under one of following conditions:

* + your metastore cluster is degraded. This is identified by following events in the CMC: OBS-ARAKOON-0007.
  + your metastore cluster has a node that is lagging behind. This is identified by following events in the CMC: OBS-ARAKOON-0008.

Executing this procedure when one or both of the previous mentioned events have triggered, will cause temporary data-unavailability.

**Why is this procedure necessary?**

To limit the time necessary to reconstruct a database, the amount of .tlf files (compressed metastore tlog files) need to be kept below 250. When the number of .tlf exceeds this number, an AmpliStor event (OBS-ARAKOON-0009) will be triggered.

In order to reduce the number of .tlf files, an operator should execute the following procedure:

In Amplistor version 3.2, the following procedure will be automated. Compliance to these instructions will reduce the time to upgrade to a newer version.

**Preparation**

**Overview:**

The process consists out of following steps:

* + Copy the .tlog and .tlf files to the metastorebackup namespace.
  + Execute a tlogcollapse.
  1. Copy the generated backup database to the metastorebackup namespace.

**Prerequisites:**

The procedure assumes that a namespace has been created that will hold a backup of all arakoon tlog files.

In case the namespace, was not yet created, you should use the CMC to create a namespace with following name:

\_metastorebackup

Use a policy for the namespace that is in line with other protection policies in the environment. The majority of files in this namespace will be bigger than 10 MB.

**Copying the .tlog and .tlf files to the designated namespace**

Proceed as follows:

* 1. Identify the metastore cluster name. This name is mentioned in the event:  
     Example  
     If the event is: "Too many tlog files (319) for node node\_0\_9005 on MetaStore objectmetadata", then the name of the metastore cluster is "***objectmetadata****".*
  2. Repeat the following steps on all members of the metastore cluster, to identify which 3 nodes make up this cluster.
     1. In the CMC, navigate to **Dashboard** >> **Administration** >> **Storage Management** >> **MetaStores**.
     2. Click on the identified metastore.
     3. Find the names of the controllers hosting the metastore members (column "**Machine**").  
        This procedure cannot be executed when a node is a master.
  3. Open an ssh connection to one of these nodes.
  4. Exit OSMI and enter the q-shell:

/opt/qbase3/qshell

* 1. Enter the following commands:
  2. replace <yourclustername> with the actual arakoon cluster name e.g. framework, env\_metastore, etc.

clusterName='<yourclustername>'

q.clients.arakoon.getClient(clusterName).whoMaster()

This will return the name of the current master node.

Since this procedure will only work on a node that is **not** the master, you can avoid this node now that its identifier is known.

* 1. From qshell use the following command to print a configuration file that can be used to correlate an node identifier with a controller IP address:

cfg = q.config.getInifile('/opt/qbase3/cfg/qconfig/arakoon/%s/%s'%(clusterName,clusterName))

print cfg.getContent()

* 1. Open an ssh connection to one of the members that is not the master.
  2. Open a screen session.

screen -S metastorebackup

This is needed since some of the commands can take a long time to complete and losing the connection might interrupt the command.

* 1. Enter qshell

/opt/qbase3/qshell

* 1. Identify the directory that contains the .tlf files:

tlogdir = q.system.fswalker.find('/mnt',pathRegexIncludes=['arakoon\_tlogs/%s'%clusterName], includeFolders=True, depths=[2])[0]

The tlog variable will contain a dir containing tlogs called <tlogdir>. It will be one of the following two:

/mnt/generic1/arakoon\_tlogs/<yourclustername>

/mnt/generic2/arakoon\_tlogs/<yourclustername>

* 1. Get arakoon arakoon local node

localNode = q.manage.arakoon.getCluster(clusterName).listLocalNodes()[0]

* 1. Create a directory structure in the \_metastorebackup namespace by entering the following commands on the shell:

cmd1 = "/opt/qbase3/bin/dss --mkdir %s -N \_metastorebackup"%tlogdir.split('/')[-1]

q.system.unix.executeAsUser(cmd1, 'root')

cmd2 = "/opt/qbase3/bin/dss --mkdir %s/%s -N \_metastorebackup"%(tlogdir.split('/')[-1], localNode)

q.system.unix.executeAsUser(cmd2, 'root')

cmd1: If the directory structure already exists, you will get following error: "**15: overwrite not allowed: directory exists**".

cmd2: Ignore the following message, if returned **(0, '\n\*\*\*ERROR\*\*\*\nstdin: is not a tty\n\n')**

* 1. Execute the following script in the same q-shell instance:

metastorecluster=tlogdir.split("/")[-1]

metastoreid=localNode

**Executing a tlog collapse**

To reduce the number of tlf files, proceed as follows:

All steps in this procedure may take several minutes to an hour, depending on the size of the metastore.

This procedure should only be done on slave Arakoon nodes. If this procedure needs to be performed on a node that is currently the master, perform a master switch. Make sure to execute the switch in a safe way (i.e. checking if slave nodes are not lagging behind prior to triggering the master switch). Check the procedure "**Performing a master switch**" below for details.

* 1. In the current active q-shell, create a backup of the current metastore.

backup\_location = q.system.fs.joinPaths(tlogdir, 'backup.db')

mycluster = q.manage.arakoon.getCluster(metastorecluster)

mycluster.backupDb(metastoreid,backup\_location)

You will see the backup.db file growing in size while the backup is taking place. You can expect to see this backup going at 100 MiB/s.

* 1. Following q-shell commands will backup your .tlf files to the  \_metastorebackup namespace.(this typically goes at a file/sec)

# First backup all tlf files, don't include \*.tlf.part files

tlflist = q.system.fswalker.find(tlogdir,pathRegexIncludes=['\.tlf$'])

tlflist.sort()

for item in tlflist:

   print item

   q.dss.client.putFile("\_metastorebackup",item.split("arakoon\_tlogs/")[1],item)

# Get highest tlog path

import os

tloglist = q.system.fswalker.find(tlogdir,pathRegexIncludes=['\.tlog$'])

highest\_tlog\_nr = 0

highest\_tlog\_path = ''

for tlog in tloglist:

   numeric = int(os.path.basename(tlog).rstrip('.tlog'))

   if numeric > highest\_tlog\_nr:

       highest\_tlog\_nr = numeric

       highest\_tlog\_path = tlog

# Now backup all tlogs that are not the latest one

for tlog in tloglist:

   if tlog != highest\_tlog\_path:

       q.dss.client.putFile("\_metastorebackup",tlog.split("arakoon\_tlogs/")[1],tlog)

* 1. In the same q-shell, run an optimize on the backup to reduce its storage consumption. (this typically runs at 10 MiB/s)

tcbmgr\_path = q.system.fs.joinPaths(q.dirs.appDir, "arakoon", "bin", "tcbmgr")

command = "%s optimize -df -nl -tl %s" % (tcbmgr\_path, backup\_location)

q.system.process.execute(command)

* 1. Now replace the current metastore database with the backup-copy and remove the .tlf files that are referred by it.

mycluster.injectAsHead(metastoreid, newHead=backup\_location)

* 1. Safeguard the backup next to the .tlf files that were backed up earlier.

backup\_name = metastorecluster + "/" + metastoreid + "/backup.db." + str(highest\_tlog\_nr)

q.dss.client.putFile("\_metastorebackup",backup\_name,backup\_location,timeout=3600)

* 1. Remove the local backup and partial tlogs.

q.system.fs.remove(backup\_location)

tlfpartlist = q.system.fswalker.find(tlogdir,pathRegexIncludes=['.tlf.part'])

for item in tlfpartlist:

    q.system.fs.remove(item)

**Performing a master switch**

To perform a master switch, proceed as follows:

* 1. Check if nodes are not lagging behind:

q.clients.arakoon.getClient('<yourclustername>').statistics()['node\_is']

This will return output, similar to the following:

{'node\_0\_9005': 50882272, 'node\_1\_9005': 50882272, 'node\_2\_9005': 50882272}

* 1. If the numbers behind the metastore node names are at most 1 number different from each other, your environment is healthy (and thus ready for a master switch).
  2. Identify the current master.

q.clients.arakoon.getClient(clusterName).whoMaster()

* 1. Login over ssh to the node running the master.
  2. Use q-shell to STOP the metastore instance on this node for a few seconds:

clusterName='<yourclustername>'

yourclustername = q.manage.arakoon.getCluster(clusterName)

yourclustername.stop()

* 1. Keep trying step 3 till another Master is elected
  2. Start the arakoon node again

yourclustername.start()

* 1. After the switch has completed, repeat step 3 to validate if another node has assumed the role of master.

Pasted from <<https://confluence.amplidata.com/display/KBEX/ARA005>>

Kind Regards

Oliver

# MetaStore framework is lagging keys on machine

2015-09-02 20:58:05,368: utilities.py: INFO: **WARNING: OBS-ARAKOON-0008**

2015-09-02 20:58:05,368: utilities.py: INFO: Node node\_1\_9003 on MetaStore framework is lagging keys on machine [controller1-la]

Events about lagging keys are common during the "collapse transaction logs" policy and will correct themselves

# DRIVE IS READ ONLY

**LATTUS-M/Blockstore '/mnt/dss/dss10/blockstore' has status DECOMMISSIONED for more than 10 days/Mountpoint '/mnt/dss/dss6' is read-only**

Auto-decommissioning a suspect bad drive, did not set to abandoned the drive

after 10 days from being decommissioned. It set the disk to DECOMMISSIONED, but may be hung at this state.

We may be seeing an issue where disks get stuck indefinitely in an auto-decommissioning state on Lattus 3.6. I have seen it with AppleMVP and I believe Zuffa has the same issue.

If you see this we  have a case open with Amplidata Q-1159 for ‘blockstores not getting abandoned’. I get the impression from Brooke at Amplidata that they know about this issue. My SR is 3583936.

This will also show itself in the Event Report as a disk has been decommissioned for 7 or 10 days.

As and when I get more concrete evidence I will forward it on. I expect something in the next day or two….

Any questions let me know

Kind Regards

Oliver

This is an existing bug. No fix available. Read only is normal when decommissioning

Stornext writing objects to that blockstore If drive is in a decommissioned state shout not be writing to that drive DECOMMISSIONED state set blockstore to read-only

**To see drive status on all the storage nodes from logs**

Ctlr01/storage\_info/Storage\_info\_2015-08\_09…

loc 15 (this is storage015)

| +- node 25: 10.224.8.23:23521, bs 144: OK, online

| +- node 25: 10.224.8.23:23521, bs 145: OK, online

| +- node 25: 10.224.8.23:23521, bs 146: OK, online

| +- node 25: 10.224.8.23:23521, bs 147: OK, online

| +- node 25: 10.224.8.23:23521, bs 148: OK, online

| +- node 25: 10.224.8.23:23521, bs 149: OK, online

| +- node 24: 10.224.8.23:23520, bs 150: OK, online

| +- node 24: 10.224.8.23:23520, bs 151: OK, online

| +- node 24: 10.224.8.23:23520, bs 152: OK, online

| +- node 24: 10.224.8.23:23520, bs 153: DECOMMISSIONED, online

| +- node 24: 10.224.8.23:23520, bs 154: OK, online

| +- node 24: 10.224.8.23:23520, bs 155: OK, online

| +- node 24: 10.224.8.23:23520, bs 433: OK, online

[**Bug 56541**](http://bo.quantum.com/bugzilla/show_bug.cgi?id=56541) - Lattus Tracker: Auto-decommissioning does not set the drive to abandoned

QUANTUM-1070

Event raised:

2015-06-02 04:48:34: Blockstore '/mnt/dss/dss6/blockstore' has status

DECOMMISSIONED for more than 10 days

storage021

| | +- loc 26

| | | +- node 40: 10.1.16.27:23520, bs 240: OK, online

| | | +- node 40: 10.1.16.27:23520, bs 241: OK, online

| | | +- node 40: 10.1.16.27:23520, bs 242: OK, online

| | | +- node 40: 10.1.16.27:23520, bs 243: OK, online

| | | +- node 40: 10.1.16.27:23520, bs 244: OK, online

| | | +- node 40: 10.1.16.27:23520, bs 245: DECOMMISSIONED, online

| | | +- node 41: 10.1.16.27:23521, bs 246: OK, online

| | | +- node 41: 10.1.16.27:23521, bs 247: OK, online

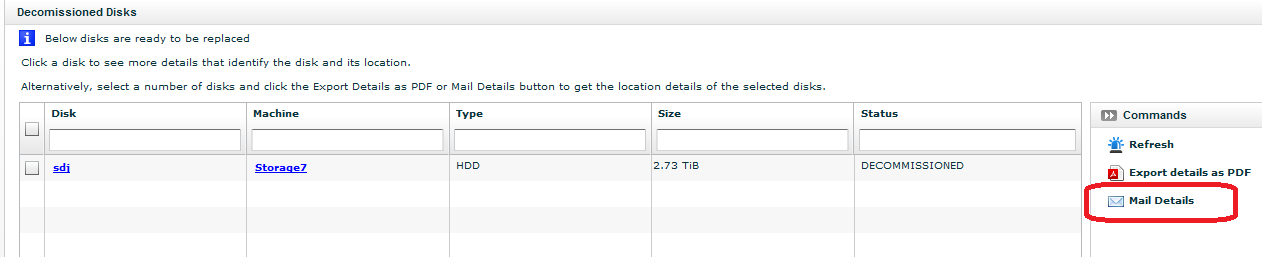
| | | +- node 41: 10.1.16.27:23521, bs 248: OK, online

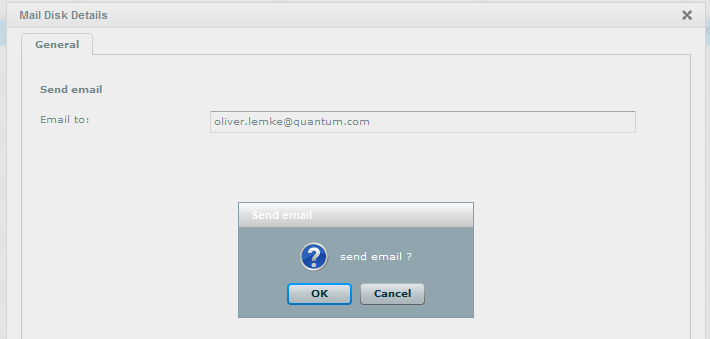
| | | +- node 41: 10.1.16.27:23521, bs 249: OK, online

| | | +- node 41: 10.1.16.27:23521, bs 250: OK, online

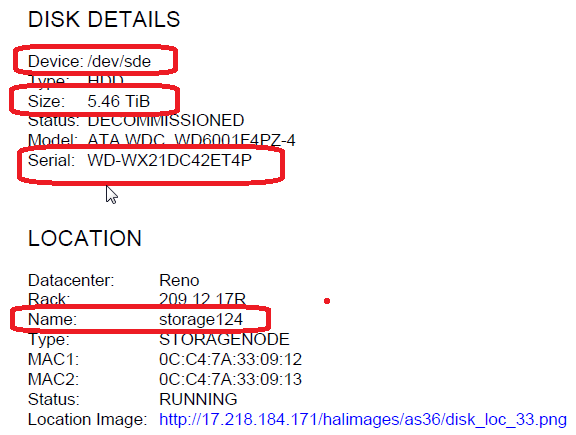
| | | +- node 41: 10.1.16.27:23521, bs 251: OK, online

If you need info of decommissioned disks, just have the customer email it directly to you form the Lattus GUI. This supposedly works on all sites that are configured for Event reports:





The report includes Device/Size/Serial # and Storage Node:



- The image link is no good, as that is local to the customer site.

# The first step after a disk goes degraded

The first step after a disk goes degraded, is to reset it via the GUI. This should be done at least once, better twice. Directly after the reset, the disk will disappear from the degraded list, but will then reappear minutes/hours/days after. You can run a diagnose on the disk which may help clarify the failure issue. Once it has returned a couple of times to the degraded list you can decommission it or let the auto-decommissioning process take care of decommissioning the drives.

Notice that if you manually decommission a drive it seems to run much quicker than the auto-decommissioning process. This is an illusion. When a disk is auto-decommissioned it will not flag as complete until all the repair agents have completed redistributing the blockstore data (each disk holds one blockstore which stays the same after the node is initialized at original setup - this then never changes). If you run a manual decommission, it just triggers the repair agents to run, but they have not completed even though the disk is marked within a few mins as decommissioned. The repair agents normally take 2-3 days on a 6TB disk.

So this now builds a list of decommissioned disks. As mentioned before we replace these under 3 categories:

1) There are 8 or more disks ready to be replaced, per site.

2) Disk safety drops significantly - dropping by 1 or 2 levels is not really significant. This will go back up once the repair agents have completed. Manually decommissioning a disk or two will not normally see an immediate effect on disk safety, you may need to wait 24 hrs to see an impact.

3) One of your disks is a RAID disk (see below). In this case we will replace the RAID disk as a priority along with any other failed disks in your decommissioned disk list, that are at the same site. This is worked on a site by site basis. There are 4 RAID disks per Storage node, 2 for metadata and 2 for O/S data.

Raid disk identification:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S10 | sda | sdb | sdc | sdd |
| S20 | sda | sdb | sdc | sdl |
| S30 | sdc | sdl | sdi | sdj |

These are typical sdX references for RAID disks and each type of Node is different. This was where I was originally confused with the sda/b/c/d, but this is only for S10 nodes. To make it a little more complicated, if two disks fail in a node, these lettering may get shifted around. You can actually click on the disk in the GUI and it will tell you if there are RAID partitions on it. The command cat /proc/mdstat will also give you that information.

RAID data is mirrored metadata or O/S data that sits alongside the storage data. Obviously if one of the mirrored disks has an issue, then we want to replace quicker to ensure that the other disks does not have time to fail also.

I hope this helps in clarifying some of the Lattus disk behavior and when we replace disks. Again for the most part, just ignore the kernel dmesg errors, unless something follows that is not disk related.

I see you are not currently setup to send us daily Lattus event reports. This I would recommend you setup, as it gives us a much better way to review your Lattus and determine if there is anything serious going on. For other customer we simply review this report and inform you if there is anything to deal with. If there is, then we open a separate case to address it. I can help you setup the Daily Event Report to go to Quantum. Note this is not the same as RAS alerts.

# Disk not detected

**LATTUS/Disk not detected: /dev/disk/by-id/ata-WDC\_WD4000F9YZ-09N20L0\_WD-WCC131641052 on environment 'Apple Computer INC** –

Need to analyze if model still has old disk or if this is new disk.

Disk not detected replace the drive

No decommissioned disks listed at all. All disks showing as OK on the node itself. I think this is a reporting issue.

# Kernel dmesg errors

**LattusKernel dmesg errors detected on environment 'Apple Computer INC - CX1421CKC00016'**

Kernel dmesg errors are pretty common to see on Lattus, as these are normally pre-cursors to disk failures. For the most part they can be ignored. You will normally find a kernel dmesg error followed either the same day or the next day by a degraded disk message. It is actually the Ubuntu Kernel that picks up the disk error, which Lattus picks up as a disk error - sometimes the kernel dmseg errors are so  minor that the disk never becomes degraded.

The first step after a disk goes degraded, is to reset it via the GUI. This should be done at least once, better twice. Directly after the reset, the disk will disappear from the degraded list, but will then reappear minutes/hours/days after. You can run a diagnose on the disk which may help clarify the failure issue. Once it has returned a couple of times to the degraded list you can decommission it or let the auto-decommissioning process take care of decommissioning the drives.

Notice that if you manually decommission a drive it seems to run much quicker than the auto-decommissioning process. This is an illusion. When a disk is auto-decommissioned it will not flag as complete until all the repair agents have completed redistributing the blockstore data (each disk holds one blockstore which stays the same after the node is initialized at original setup - this then never changes). If you run a manual decommission, it just triggers the repair agents to run, but they have not completed even though the disk is marked within a few mins as decommissioned. The repair agents normally take 2-3 days on a 6TB disk.

So this now builds a list of decommissioned disks. As mentioned before we replace these under 3 categories:

1) There are 8 or more disks ready to be replaced, per site.

2) Disk safety drops significantly - dropping by 1 or 2 levels is not really significant. This will go back up once the repair agents have completed. Manually decommissioning a disk or two will not normally see an immediate effect on disk safety, you may need to wait 24 hrs to see an impact.

3) One of your disks is a RAID disk (see below). In this case we will replace the RAID disk as a priority along with any other failed disks in your decommissioned disk list, that are at the same site. This is worked on a site by site basis. There are 4 RAID disks per Storage node, 2 for metadata and 2 for O/S data.

Raid disk identification:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S10 | sda | sdb | sdc | sdd |
| S20 | sda | sdb | sdc | sdl |
| S30 | sdc | sdl | sdi | sdj |

These are typical sdX references for RAID disks and each type of Node is different. This was where I was originally confused with the sda/b/c/d, but this is only for S10 nodes. To make it a little more complicated, if two disks fail in a node, these lettering may get shifted around. You can actually click on the disk in the GUI and it will tell you if there are RAID partitions on it. The command cat /proc/mdstat will also give you that information.

RAID data is mirrored metadata or O/S data that sits alongside the storage data. Obviously if one of the mirrored disks has an issue, then we want to replace quicker to ensure that the other disks does not have time to fail also.

I hope this helps in clarifying some of the Lattus disk behavior and when we replace disks. Again for the most part, just ignore the kernel dmesg errors, unless something follows that is not disk related.

I see you are not currently setup to send us daily Lattus event reports. This I would recommend you setup, as it gives us a much better way to review your Lattus and determine if there is anything serious going on. For other customer we simply review this report and inform you if there is anything to deal with. If there is, then we open a separate case to address it. I can help you setup the Daily Event Report to go to Quantum. Note this is not the same as RAS alerts.

File system errors after disk came back up. Suggest replacing the associated drives.

Check /var/log/kern.log

Example:

Oct 20 09:35:01 localhost kernel: [1029170.973020] sas2ircu: page allocation

failure: order:4, mode:0x8020

The kernel error message gives order:4 and mode:0x8020. The order argument is the log of the number of pages requested from the allocator, (e.g. order is 1 for 2 page requests, or 4 when 16 pages are requested. Therefore the error occurred when requesting 16 pages. The mode argument specifies the gfp\_flags that were passed with the allocation request. These flags specify the requested behavior of the memory allocator (e.g. GFP\_KERNEL, GFP\_ATOMIC, ...).

**LATTUS BUG EXAMPLES WITH KERNEL DEBUG ERRORS:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| [**ID**](http://bo.quantum.com/bugzilla/buglist.cgi?bug_status=REOPENED&bug_status=OBSERVE&bug_status=NEW&bug_status=ASSIGNED&field0-0-0=product&field0-0-1=component&field0-0-2=alias&field0-0-3=short_desc&field0-0-4=status_whiteboard&field0-0-5=content&field1-0-0=product&field1-0-1=component&field1-0-2=alias&field1-0-3=short_desc&field1-0-4=status_whiteboard&field1-0-5=content&field2-0-0=product&field2-0-1=component&field2-0-2=alias&field2-0-3=short_desc&field2-0-4=status_whiteboard&field2-0-5=content&field3-0-0=product&field3-0-1=component&field3-0-2=alias&field3-0-3=short_desc&field3-0-4=status_whiteboard&field3-0-5=content&query_format=advanced&type0-0-0=substring&type0-0-1=substring&type0-0-2=substring&type0-0-3=substring&type0-0-4=substring&type0-0-5=matches&type1-0-0=substring&type1-0-1=substring&type1-0-2=substring&type1-0-3=substring&type1-0-4=substring&type1-0-5=matches&type2-0-0=substring&type2-0-1=substring&type2-0-2=substring&type2-0-3=substring&type2-0-4=substring&type2-0-5=matches&type3-0-0=substring&type3-0-1=substring&type3-0-2=substring&type3-0-3=substring&type3-0-4=substring&type3-0-5=matches&value0-0-0=Lattus&value0-0-1=Lattus&value0-0-2=Lattus&value0-0-3=Lattus&value0-0-4=Lattus&value0-0-5=%22Lattus%22&value1-0-0=Kernel&value1-0-1=Kernel&value1-0-2=Kernel&value1-0-3=Kernel&value1-0-4=Kernel&value1-0-5=%22Kernel%22&value2-0-0=dmesg&value2-0-1=dmesg&value2-0-2=dmesg&value2-0-3=dmesg&value2-0-4=dmesg&value2-0-5=%22dmesg%22&value3-0-0=errors&value3-0-1=errors&value3-0-2=errors&value3-0-3=errors&value3-0-4=errors&value3-0-5=%22errors%22&order=bug_id&query_based_on=) | [**Sev**](http://bo.quantum.com/bugzilla/buglist.cgi?bug_status=REOPENED&bug_status=OBSERVE&bug_status=NEW&bug_status=ASSIGNED&field0-0-0=product&field0-0-1=component&field0-0-2=alias&field0-0-3=short_desc&field0-0-4=status_whiteboard&field0-0-5=content&field1-0-0=product&field1-0-1=component&field1-0-2=alias&field1-0-3=short_desc&field1-0-4=status_whiteboard&field1-0-5=content&field2-0-0=product&field2-0-1=component&field2-0-2=alias&field2-0-3=short_desc&field2-0-4=status_whiteboard&field2-0-5=content&field3-0-0=product&field3-0-1=component&field3-0-2=alias&field3-0-3=short_desc&field3-0-4=status_whiteboard&field3-0-5=content&query_format=advanced&type0-0-0=substring&type0-0-1=substring&type0-0-2=substring&type0-0-3=substring&type0-0-4=substring&type0-0-5=matches&type1-0-0=substring&type1-0-1=substring&type1-0-2=substring&type1-0-3=substring&type1-0-4=substring&type1-0-5=matches&type2-0-0=substring&type2-0-1=substring&type2-0-2=substring&type2-0-3=substring&type2-0-4=substring&type2-0-5=matches&type3-0-0=substring&type3-0-1=substring&type3-0-2=substring&type3-0-3=substring&type3-0-4=substring&type3-0-5=matches&value0-0-0=Lattus&value0-0-1=Lattus&value0-0-2=Lattus&value0-0-3=Lattus&value0-0-4=Lattus&value0-0-5=%22Lattus%22&value1-0-0=Kernel&value1-0-1=Kernel&value1-0-2=Kernel&value1-0-3=Kernel&value1-0-4=Kernel&value1-0-5=%22Kernel%22&value2-0-0=dmesg&value2-0-1=dmesg&value2-0-2=dmesg&value2-0-3=dmesg&value2-0-4=dmesg&value2-0-5=%22dmesg%22&value3-0-0=errors&value3-0-1=errors&value3-0-2=errors&value3-0-3=errors&value3-0-4=errors&value3-0-5=%22errors%22&order=bug_severity%2Crelevance%20DESC&query_based_on=) | [**Pri**](http://bo.quantum.com/bugzilla/buglist.cgi?bug_status=REOPENED&bug_status=OBSERVE&bug_status=NEW&bug_status=ASSIGNED&field0-0-0=product&field0-0-1=component&field0-0-2=alias&field0-0-3=short_desc&field0-0-4=status_whiteboard&field0-0-5=content&field1-0-0=product&field1-0-1=component&field1-0-2=alias&field1-0-3=short_desc&field1-0-4=status_whiteboard&field1-0-5=content&field2-0-0=product&field2-0-1=component&field2-0-2=alias&field2-0-3=short_desc&field2-0-4=status_whiteboard&field2-0-5=content&field3-0-0=product&field3-0-1=component&field3-0-2=alias&field3-0-3=short_desc&field3-0-4=status_whiteboard&field3-0-5=content&query_format=advanced&type0-0-0=substring&type0-0-1=substring&type0-0-2=substring&type0-0-3=substring&type0-0-4=substring&type0-0-5=matches&type1-0-0=substring&type1-0-1=substring&type1-0-2=substring&type1-0-3=substring&type1-0-4=substring&type1-0-5=matches&type2-0-0=substring&type2-0-1=substring&type2-0-2=substring&type2-0-3=substring&type2-0-4=substring&type2-0-5=matches&type3-0-0=substring&type3-0-1=substring&type3-0-2=substring&type3-0-3=substring&type3-0-4=substring&type3-0-5=matches&value0-0-0=Lattus&value0-0-1=Lattus&value0-0-2=Lattus&value0-0-3=Lattus&value0-0-4=Lattus&value0-0-5=%22Lattus%22&value1-0-0=Kernel&value1-0-1=Kernel&value1-0-2=Kernel&value1-0-3=Kernel&value1-0-4=Kernel&value1-0-5=%22Kernel%22&value2-0-0=dmesg&value2-0-1=dmesg&value2-0-2=dmesg&value2-0-3=dmesg&value2-0-4=dmesg&value2-0-5=%22dmesg%22&value3-0-0=errors&value3-0-1=errors&value3-0-2=errors&value3-0-3=errors&value3-0-4=errors&value3-0-5=%22errors%22&order=priority%2Crelevance%20DESC&query_based_on=) | [**OS**](http://bo.quantum.com/bugzilla/buglist.cgi?bug_status=REOPENED&bug_status=OBSERVE&bug_status=NEW&bug_status=ASSIGNED&field0-0-0=product&field0-0-1=component&field0-0-2=alias&field0-0-3=short_desc&field0-0-4=status_whiteboard&field0-0-5=content&field1-0-0=product&field1-0-1=component&field1-0-2=alias&field1-0-3=short_desc&field1-0-4=status_whiteboard&field1-0-5=content&field2-0-0=product&field2-0-1=component&field2-0-2=alias&field2-0-3=short_desc&field2-0-4=status_whiteboard&field2-0-5=content&field3-0-0=product&field3-0-1=component&field3-0-2=alias&field3-0-3=short_desc&field3-0-4=status_whiteboard&field3-0-5=content&query_format=advanced&type0-0-0=substring&type0-0-1=substring&type0-0-2=substring&type0-0-3=substring&type0-0-4=substring&type0-0-5=matches&type1-0-0=substring&type1-0-1=substring&type1-0-2=substring&type1-0-3=substring&type1-0-4=substring&type1-0-5=matches&type2-0-0=substring&type2-0-1=substring&type2-0-2=substring&type2-0-3=substring&type2-0-4=substring&type2-0-5=matches&type3-0-0=substring&type3-0-1=substring&type3-0-2=substring&type3-0-3=substring&type3-0-4=substring&type3-0-5=matches&value0-0-0=Lattus&value0-0-1=Lattus&value0-0-2=Lattus&value0-0-3=Lattus&value0-0-4=Lattus&value0-0-5=%22Lattus%22&value1-0-0=Kernel&value1-0-1=Kernel&value1-0-2=Kernel&value1-0-3=Kernel&value1-0-4=Kernel&value1-0-5=%22Kernel%22&value2-0-0=dmesg&value2-0-1=dmesg&value2-0-2=dmesg&value2-0-3=dmesg&value2-0-4=dmesg&value2-0-5=%22dmesg%22&value3-0-0=errors&value3-0-1=errors&value3-0-2=errors&value3-0-3=errors&value3-0-4=errors&value3-0-5=%22errors%22&order=op_sys%2Crelevance%20DESC&query_based_on=) | [**Assignee**](http://bo.quantum.com/bugzilla/buglist.cgi?bug_status=REOPENED&bug_status=OBSERVE&bug_status=NEW&bug_status=ASSIGNED&field0-0-0=product&field0-0-1=component&field0-0-2=alias&field0-0-3=short_desc&field0-0-4=status_whiteboard&field0-0-5=content&field1-0-0=product&field1-0-1=component&field1-0-2=alias&field1-0-3=short_desc&field1-0-4=status_whiteboard&field1-0-5=content&field2-0-0=product&field2-0-1=component&field2-0-2=alias&field2-0-3=short_desc&field2-0-4=status_whiteboard&field2-0-5=content&field3-0-0=product&field3-0-1=component&field3-0-2=alias&field3-0-3=short_desc&field3-0-4=status_whiteboard&field3-0-5=content&query_format=advanced&type0-0-0=substring&type0-0-1=substring&type0-0-2=substring&type0-0-3=substring&type0-0-4=substring&type0-0-5=matches&type1-0-0=substring&type1-0-1=substring&type1-0-2=substring&type1-0-3=substring&type1-0-4=substring&type1-0-5=matches&type2-0-0=substring&type2-0-1=substring&type2-0-2=substring&type2-0-3=substring&type2-0-4=substring&type2-0-5=matches&type3-0-0=substring&type3-0-1=substring&type3-0-2=substring&type3-0-3=substring&type3-0-4=substring&type3-0-5=matches&value0-0-0=Lattus&value0-0-1=Lattus&value0-0-2=Lattus&value0-0-3=Lattus&value0-0-4=Lattus&value0-0-5=%22Lattus%22&value1-0-0=Kernel&value1-0-1=Kernel&value1-0-2=Kernel&value1-0-3=Kernel&value1-0-4=Kernel&value1-0-5=%22Kernel%22&value2-0-0=dmesg&value2-0-1=dmesg&value2-0-2=dmesg&value2-0-3=dmesg&value2-0-4=dmesg&value2-0-5=%22dmesg%22&value3-0-0=errors&value3-0-1=errors&value3-0-2=errors&value3-0-3=errors&value3-0-4=errors&value3-0-5=%22errors%22&order=assigned_to%2Crelevance%20DESC&query_based_on=) | [**Status**](http://bo.quantum.com/bugzilla/buglist.cgi?bug_status=REOPENED&bug_status=OBSERVE&bug_status=NEW&bug_status=ASSIGNED&field0-0-0=product&field0-0-1=component&field0-0-2=alias&field0-0-3=short_desc&field0-0-4=status_whiteboard&field0-0-5=content&field1-0-0=product&field1-0-1=component&field1-0-2=alias&field1-0-3=short_desc&field1-0-4=status_whiteboard&field1-0-5=content&field2-0-0=product&field2-0-1=component&field2-0-2=alias&field2-0-3=short_desc&field2-0-4=status_whiteboard&field2-0-5=content&field3-0-0=product&field3-0-1=component&field3-0-2=alias&field3-0-3=short_desc&field3-0-4=status_whiteboard&field3-0-5=content&query_format=advanced&type0-0-0=substring&type0-0-1=substring&type0-0-2=substring&type0-0-3=substring&type0-0-4=substring&type0-0-5=matches&type1-0-0=substring&type1-0-1=substring&type1-0-2=substring&type1-0-3=substring&type1-0-4=substring&type1-0-5=matches&type2-0-0=substring&type2-0-1=substring&type2-0-2=substring&type2-0-3=substring&type2-0-4=substring&type2-0-5=matches&type3-0-0=substring&type3-0-1=substring&type3-0-2=substring&type3-0-3=substring&type3-0-4=substring&type3-0-5=matches&value0-0-0=Lattus&value0-0-1=Lattus&value0-0-2=Lattus&value0-0-3=Lattus&value0-0-4=Lattus&value0-0-5=%22Lattus%22&value1-0-0=Kernel&value1-0-1=Kernel&value1-0-2=Kernel&value1-0-3=Kernel&value1-0-4=Kernel&value1-0-5=%22Kernel%22&value2-0-0=dmesg&value2-0-1=dmesg&value2-0-2=dmesg&value2-0-3=dmesg&value2-0-4=dmesg&value2-0-5=%22dmesg%22&value3-0-0=errors&value3-0-1=errors&value3-0-2=errors&value3-0-3=errors&value3-0-4=errors&value3-0-5=%22errors%22&order=bug_status%2Crelevance%20DESC&query_based_on=) | [**Resolution**](http://bo.quantum.com/bugzilla/buglist.cgi?bug_status=REOPENED&bug_status=OBSERVE&bug_status=NEW&bug_status=ASSIGNED&field0-0-0=product&field0-0-1=component&field0-0-2=alias&field0-0-3=short_desc&field0-0-4=status_whiteboard&field0-0-5=content&field1-0-0=product&field1-0-1=component&field1-0-2=alias&field1-0-3=short_desc&field1-0-4=status_whiteboard&field1-0-5=content&field2-0-0=product&field2-0-1=component&field2-0-2=alias&field2-0-3=short_desc&field2-0-4=status_whiteboard&field2-0-5=content&field3-0-0=product&field3-0-1=component&field3-0-2=alias&field3-0-3=short_desc&field3-0-4=status_whiteboard&field3-0-5=content&query_format=advanced&type0-0-0=substring&type0-0-1=substring&type0-0-2=substring&type0-0-3=substring&type0-0-4=substring&type0-0-5=matches&type1-0-0=substring&type1-0-1=substring&type1-0-2=substring&type1-0-3=substring&type1-0-4=substring&type1-0-5=matches&type2-0-0=substring&type2-0-1=substring&type2-0-2=substring&type2-0-3=substring&type2-0-4=substring&type2-0-5=matches&type3-0-0=substring&type3-0-1=substring&type3-0-2=substring&type3-0-3=substring&type3-0-4=substring&type3-0-5=matches&value0-0-0=Lattus&value0-0-1=Lattus&value0-0-2=Lattus&value0-0-3=Lattus&value0-0-4=Lattus&value0-0-5=%22Lattus%22&value1-0-0=Kernel&value1-0-1=Kernel&value1-0-2=Kernel&value1-0-3=Kernel&value1-0-4=Kernel&value1-0-5=%22Kernel%22&value2-0-0=dmesg&value2-0-1=dmesg&value2-0-2=dmesg&value2-0-3=dmesg&value2-0-4=dmesg&value2-0-5=%22dmesg%22&value3-0-0=errors&value3-0-1=errors&value3-0-2=errors&value3-0-3=errors&value3-0-4=errors&value3-0-5=%22errors%22&order=resolution%2Crelevance%20DESC&query_based_on=) | [**Summary**](http://bo.quantum.com/bugzilla/buglist.cgi?bug_status=REOPENED&bug_status=OBSERVE&bug_status=NEW&bug_status=ASSIGNED&field0-0-0=product&field0-0-1=component&field0-0-2=alias&field0-0-3=short_desc&field0-0-4=status_whiteboard&field0-0-5=content&field1-0-0=product&field1-0-1=component&field1-0-2=alias&field1-0-3=short_desc&field1-0-4=status_whiteboard&field1-0-5=content&field2-0-0=product&field2-0-1=component&field2-0-2=alias&field2-0-3=short_desc&field2-0-4=status_whiteboard&field2-0-5=content&field3-0-0=product&field3-0-1=component&field3-0-2=alias&field3-0-3=short_desc&field3-0-4=status_whiteboard&field3-0-5=content&query_format=advanced&type0-0-0=substring&type0-0-1=substring&type0-0-2=substring&type0-0-3=substring&type0-0-4=substring&type0-0-5=matches&type1-0-0=substring&type1-0-1=substring&type1-0-2=substring&type1-0-3=substring&type1-0-4=substring&type1-0-5=matches&type2-0-0=substring&type2-0-1=substring&type2-0-2=substring&type2-0-3=substring&type2-0-4=substring&type2-0-5=matches&type3-0-0=substring&type3-0-1=substring&type3-0-2=substring&type3-0-3=substring&type3-0-4=substring&type3-0-5=matches&value0-0-0=Lattus&value0-0-1=Lattus&value0-0-2=Lattus&value0-0-3=Lattus&value0-0-4=Lattus&value0-0-5=%22Lattus%22&value1-0-0=Kernel&value1-0-1=Kernel&value1-0-2=Kernel&value1-0-3=Kernel&value1-0-4=Kernel&value1-0-5=%22Kernel%22&value2-0-0=dmesg&value2-0-1=dmesg&value2-0-2=dmesg&value2-0-3=dmesg&value2-0-4=dmesg&value2-0-5=%22dmesg%22&value3-0-0=errors&value3-0-1=errors&value3-0-2=errors&value3-0-3=errors&value3-0-4=errors&value3-0-5=%22errors%22&order=short_desc%2Crelevance%20DESC&query_based_on=) |
| [47531](http://bo.quantum.com/bugzilla/show_bug.cgi?id=47531) | min | P3 | All | homer.simpson | REOP | --- | [Amplidata: IPMI not configured triggers kernel dmesg events](http://bo.quantum.com/bugzilla/show_bug.cgi?id=47531) |
| [53221](http://bo.quantum.com/bugzilla/show_bug.cgi?id=53221) | cri | P3 | All | homer.simpson | REOP | --- | [BUG: unable to handle kernel NULL pointer dereference at 000000000000001c](http://bo.quantum.com/bugzilla/show_bug.cgi?id=53221) |
| [46512](http://bo.quantum.com/bugzilla/show_bug.cgi?id=46512) | maj | P3 | All | homer.simpson | NEW | --- | [AS30 storage node : dss.bin[16152]: segfault at 8 ip 00007f778108a7cf sp 00007fffaef97fc0 error 4 in libc-2.16.so](http://bo.quantum.com/bugzilla/show_bug.cgi?id=46512) |
| [51594](http://bo.quantum.com/bugzilla/show_bug.cgi?id=51594) | maj | P3 | All | homer.simpson | NEW | --- | [Lattus Tracker: Initialize of GEO: initialize fails due to error code 1 on MGMT node](http://bo.quantum.com/bugzilla/show_bug.cgi?id=51594) |
| [55445](http://bo.quantum.com/bugzilla/show_bug.cgi?id=55445) | maj | P3 | All | steve.ino | NEW | --- | [Unmounting partitions error during decommission job. Device is busy](http://bo.quantum.com/bugzilla/show_bug.cgi?id=55445) |
| [51475](http://bo.quantum.com/bugzilla/show_bug.cgi?id=51475) | cri | P3 | All | john.kastner | REOP | --- | [NDS1121 : sas2ircu: page allocation failure: order:4,](http://bo.quantum.com/bugzilla/show_bug.cgi?id=51475) |

# EXAMPLE OF DISK FAILURES PENDING VIA Kernel dmesg errors

Possibly more 6TB disk failures pending:  
Sunnyvale:  
storage040 ERROR 1 9/13/2015 10:55:21 9/13/2015 10:55:21 Kernel dmesg errors detected  
storage041 ERROR 1 9/13/2015 2:52:12 9/13/2015 2:52:12 Kernel dmesg errors detected  
storage052 ERROR 1 9/13/2015 10:39:36 9/13/2015 10:39:36 Kernel dmesg errors detected  
storage060 ERROR 1 9/13/2015 12:15:46 9/13/2015 12:15:46 Kernel dmesg errors detected

Reno:  
storage104 ERROR 1 9/12/2015 17:32:39 9/12/2015 17:32:39 Kernel dmesg errors detected  
storage108 ERROR 1 9/12/2015 17:58:03 9/12/2015 17:58:03 Kernel dmesg errors detected  
storage115 ERROR 1 9/13/2015 11:40:22 9/13/2015 11:40:22 Kernel dmesg errors detected  
storage120 ERROR 1 9/12/2015 17:28:17 9/12/2015 17:28:17 Kernel dmesg errors detected - disk 120/sda already flagged as degraded.  
storage122 ERROR 1 9/13/2015 11:37:34 9/13/2015 11:37:34 Kernel dmesg errors detected  
storage135 ERROR 1 9/12/2015 17:31:21 9/12/2015 17:31:21 Kernel dmesg errors detected

RenoDR:  
storage186 WARNING 4 9/13/2015 11:15:37 9/13/2015 12:05:41 Software RAID array /dev/md2 has status clean, degraded  
storage187 ERROR 1 9/13/2015 11:36:28 9/13/2015 11:36:28 Kernel dmesg errors detected  
storage205 ERROR 1 9/13/2015 12:05:16 9/13/2015 12:05:16 Kernel dmesg errors detected  
storage209 ERROR 1 9/12/2015 14:40:54 9/12/2015 14:40:54 Kernel dmesg errors detected

Quantum-1168

# Machine was rebooted

**LATTUS-M/Machine was rebooted**

Informational msg.

# Disk failures for

**Lattus / Disk failures for sda on environment 'Apple Computer INC - CX1421CKC00016'**

Don’t know if controller or storage node. Need to identify which drive it was.

It is not 100% certain… but as a rule of thumb…

More research needs to be done to determine if this is hard coded or not.

S10 and S20 raids are on sda, sdb, sdc, and sdd

md2 : active raid1 sdc1[1] sdd1[0]

      52395904 blocks super 1.2 [2/2] [UU]

md0 : active raid1 sda2[0] sdb2[1]

      26197888 blocks super 1.2 [2/2] [UU]

md5 : active raid1 sdb3[1] sda3[0]

      26197888 blocks super 1.2 [2/2] [UU]

md1 : active raid1 sdb4[1] sda4[0]

      2096064 blocks super 1.2 [2/2] [UU]

S20model2 (have not checked)

S30 raids are on sdc, sdi, sdj and sdl

md5 : active raid1 sdj3[1] sdi3[0]

      26197888 blocks super 1.2 [2/2] [UU]

md1 : active raid1 sdi4[0] sdj4[1]

      2096064 blocks super 1.2 [2/2] [UU]

md0 : active raid1 sdj2[1] sdi2[0]

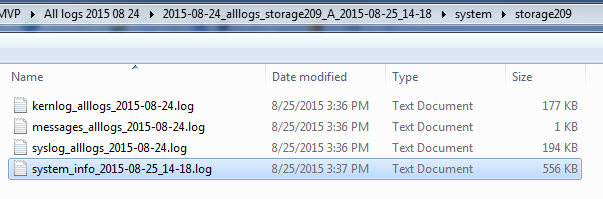
      26197888 blocks super 1.2 [2/2] [UU]

md2 : active raid1 sdl1[0] sdc1[1]

      52395904 blocks super 1.2 [2/2] [UU]

Matt Taylor | Software Product Support | Quantum | 303.588.1713 | [matt.taylor@quantum.com](mailto:matt.taylor@quantum.com)

Lattus O/S and Metadata RAID disks need immediate attention re disk replacement but non RAID do not. The question is which disks are which? I was originally under the impression that /dev/sda/sdb/sdc/sdc were the four RAID disks, but this is not the case!!! The command: ‘#cat /proc/mdstat‘ is the easiest check, but as we encourage customers to not have CLI root access, then where do we get the info. The answer is in the system\_info log under each Storage Node:



Open this log and search for the above command. You will notice 3 consecutive commands that give you all the info you need:

#df -h

#cat /proc/mounts

#cat /proc/mdstat

**Note the correlations between the /dev/sdN and /mont/dss/dssN:**

#df -h

Filesystem      Size  Used Avail Use% Mounted on

/dev/md0         25G  2.1G   22G   9% /

udev            3.9G  4.0K  3.9G   1% /dev

tmpfs           1.6G  408K  1.6G   1% /run

none            5.0M     0  5.0M   0% /run/lock

none            3.9G     0  3.9G   0% /run/shm

/dev/md2         50G  2.7G   44G   6% /mnt/sandboxtmp

/dev/sdb1       5.5T  2.1T  3.4T  38% /mnt/dss/dss5

/dev/sdd1       5.5T  2.1T  3.4T  38% /mnt/dss/dss11

/dev/sdk1       5.5T  2.1T  3.4T  38% /mnt/dss/dss7

/dev/sdg1       5.5T  2.1T  3.4T  38% /mnt/dss/dss10

/dev/sdh1       5.5T  2.1T  3.4T  38% /mnt/dss/dss8

/dev/sde1       5.5T  2.0T  3.4T  37% /mnt/dss/dss12

/dev/sdf1       5.5T  2.1T  3.4T  38% /mnt/dss/dss9

/dev/sda1       5.5T  2.1T  3.4T  38% /mnt/dss/dss6

/dev/sdl2       5.4T  2.0T  3.4T  38% /mnt/dss/dss4

/dev/sdc2       5.4T  2.0T  3.4T  38% /mnt/dss/dss3

/dev/sdi5       5.4T  2.0T  3.4T  38% /mnt/dss/dss2

/dev/sdj5       5.4T  2.0T  3.4T  38% /mnt/dss/dss1

#cat /proc/mounts

rootfs / rootfs rw 0 0

sysfs /sys sysfs rw,nosuid,nodev,noexec,relatime 0 0

proc /proc proc rw,nosuid,nodev,noexec,relatime 0 0

udev /dev devtmpfs rw,relatime,size=4055296k,nr\_inodes=1013824,mode=755 0 0

devpts /dev/pts devpts rw,nosuid,noexec,relatime,gid=5,mode=620,ptmxmode=000 0 0

tmpfs /run tmpfs rw,nosuid,relatime,size=1626984k,mode=755 0 0

/dev/disk/by-uuid/107ab1b9-584d-470f-8125-229a9858a67d / ext4 rw,relatime,errors=remount-ro,data=ordered 0 0

none /sys/fs/fuse/connections fusectl rw,relatime 0 0

none /sys/kernel/debug debugfs rw,relatime 0 0

none /sys/kernel/security securityfs rw,relatime 0 0

none /run/lock tmpfs rw,nosuid,nodev,noexec,relatime,size=5120k 0 0

none /run/shm tmpfs rw,nosuid,nodev,relatime 0 0

/dev/md2 /mnt/sandboxtmp ext4 rw,noatime,data=ordered 0 0

/dev/sdb1 /mnt/dss/dss5 ext4 rw,noatime,data=ordered 0 0

/dev/sdd1 /mnt/dss/dss11 ext4 rw,noatime,data=ordered 0 0

rpc\_pipefs /run/rpc\_pipefs rpc\_pipefs rw,relatime 0 0

/dev/sdk1 /mnt/dss/dss7 ext4 rw,noatime,data=ordered 0 0

/dev/sdg1 /mnt/dss/dss10 ext4 rw,noatime,data=ordered 0 0

/dev/sdh1 /mnt/dss/dss8 ext4 rw,noatime,data=ordered 0 0

/dev/sde1 /mnt/dss/dss12 ext4 ro,noatime,data=ordered 0 0

/dev/sdf1 /mnt/dss/dss9 ext4 rw,noatime,data=ordered 0 0

/dev/sda1 /mnt/dss/dss6 ext4 rw,noatime,data=ordered 0 0

/dev/sdl2 /mnt/dss/dss4 ext4 rw,noatime,data=ordered 0 0

/dev/sdc2 /mnt/dss/dss3 ext4 rw,noatime,data=ordered 0 0

/dev/sdi5 /mnt/dss/dss2 ext4 rw,noatime,data=ordered 0 0

/dev/sdj5 /mnt/dss/dss1 ext4 rw,noatime,data=ordered 0 0

#cat /proc/mdstat

Personalities : [raid1] [linear] [multipath] [raid0] [raid6] [raid5] [raid4] [raid10]

md5 : active raid1 sdj3[1] sdi3[0] >> dss1, dss2

      26197888 blocks super 1.2 [2/2] [UU]

md0 : active raid1 sdi2[0] sdj2[1] >> dss1, dss2

      26197888 blocks super 1.2 [2/2] [UU]

md2 : active raid1 sdl1[0] sdc1[1] >> dss4, dss3

      52395904 blocks super 1.2 [2/2] [UU]

md1 : active raid1 sdj4[1] sdi4[0] >> dss1, dss2

      2096064 blocks super 1.2 [2/2] [UU]

I am still trying to confirm if the dss# relate directly to the disk numbering in a node:

Machine generated alternative text: -
Front
.

|  |  |  |  |
| --- | --- | --- | --- |
| 9 | 10 | 11 | 12 |
| 5 | 6 | 7 | 8 |
| 1 | 2 | 3 | 4 |

   ########### Front ###########

Hope this helps clear up some of the mystery.

Kind Regards

Oliver

# Blockstore temporarily offline

**LATTUS / You have 11 blockstore(s)that is(are) temporarily offline. Blockstores are only in this status**

when some of your storage nodes are down. on environment 'Apple Computer INC - CX1421CKC00016'

Need to identify what blockstores. Could be blacklist, could be contributing to autodecommissioning issue

Example of blacklist message:

Feb 27 10:42:52.9780 warning [1027] [sb 0] could not generate complete spread:generated spread: [[109, 134, 96, 82, 84, 127, 164, 189, 199, 151, 173, 216]]

Feb 27 10:42:55.9862 info [1027] [sb 0] BS registry: BS 216 at 10.6.0.8:23521:

**incrementing blacklist counter** because of Fatal:could not connect to server at 10.6.0.8:23521

[**Bug 47294**](http://bo.quantum.com/bugzilla/show_bug.cgi?id=47294) - Amplidata Tracker: 3.4.2 rack shutdown and power-on produces RAS alerts blockstores are OFFLINE

# Low disk safeties

**Lattus/Objects found with low disk safeties on environment 'Apple Computer INC - CX1421CKC00016'**

Need to identify details of disk safety. This is an old message.

The disk safety is the spread number of blockstores that can be lost and still provide safe recovery of your data.

The CMC View: Worst case overall disk safety

Not always up-to-date, best method is verification using the qshell.

**q.dss.manage.monitorNameSpace('data01',timeout=3600)**

Identify the following:

**disksafety\_objects**

disksafety that has corrupted or is missing check blocks, fields are displayed in disk (safety:nr\_of\_objects).

'disksafety\_objects': {3: 2093, 4: 35101},

**disksafety\_objects\_offline**

The values include disk\_safety objects + disk\_safety\_objects\_offline, the safety of objects can be affected by disks in the offline state.

'disksafety\_objects\_offline': {3: 2093, 4: 35101}

**diskafety\_objects\_decommissioned**

The values include disk\_safety objects + disk\_safety\_objects\_offline + disksafety\_objects\_decommissioned

'disksafety\_objects\_decommissioned': {3: 209 , 4: 35101},

**nr\_objects\_repair**

The value includes objects that are pending writes to the metastore, where there is a difference between totals of object optimal disk safety. This is the case where the safety is less than configured disk safety

nr\_objects\_repair': 2093

[**Bug 47394**](http://bo.quantum.com/bugzilla/show_bug.cgi?id=47394) - "Objects found with low disk safeties" events after storage node reboot

Possible solution:

Multiple disks have failed and repair has not been able to correct this.

Validate that repair is ongoing and moving in the right direction, i.e. leading

to an increased disksafety.

There are mulitple issues with this situation:

1. The event solution states that "Multiple disks have failed". However in

this situation no disks have failed. A storage node was simply shutdown and

rebooted.

2. It appears that the repair crawl is either:

A. not automatically running every 24 hours, or

B. failing to repair the low disk safety when it does run

3. Critical severity for this event does not seem appropriate until a repair

crawl has occurred.

Will upload logs to evidence directory with the name of this ticket

\*WORKAROUNDS\*

Use OSMI to manually repair all namespaces

The following has been deduced from the logs:

1- While the node was shutdown there were puts happening.

this can be verified by grepping "put: Namespace" on the clientdaemon logs at

the time of the shutdown, i can see lots of puts happening on Mar 26

2- The policy used is a 12/8, with DynamicSafety, so it makes sense there were

objects written with lower disksafety

3- Crawling does happen every 24 hours keep in mind the size of the objects and

their number the repairs can take more than one day and is configured so by

check the storagedaemon logs

here is an example from storage daemon 12, m660\_copy's master daemon

$ grep "repair manager: repair crawl: namespace"

dssstoragedaemons/geo-storage-9/alllogs/geo-storage-9\_alllogs\_16\_F\_2014-03-26\_T\_2014-03-28.log

Mar 26 11:00:10.7204 info [None] repair manager: repair crawl: namespace

m660\_copy3: processed 100000 objects, 0 tasks created

Mar 26 11:00:21.3115 info [None] repair manager: repair crawl: namespace

m660\_copy3: processed 200000 objects, 0 tasks created

Mar 26 11:00:31.0714 info [None] repair manager: repair crawl: namespace

m660\_copy3: processed 300000 objects, 0 tasks created

Mar 26 11:05:45.2370 info [None] repair manager: repair crawl: namespace

m660\_copy3: processed 363281 objects, 0 tasks created

Mar 27 19:46:28.1548 info [None] repair manager: repair crawl: namespace

m660\_copy3: processed 626478 objects, 17997 tasks created

4- Regarding the "multiple disks have failed" message is merely a suggestion to

what might of gone wrong. As mo

# [storageXXX] needs to be diagnosed and replaced

**Lattus / Disk [/dev/disk/by-id/ata-WDC\_WD6001F4PZ-49ZWCM0\_WD-WX31DB48XT5R] on machine [storage209] needs to be diagnosed and replaced if necessary on environment 'Apple Computer INC - CX1421CKC00016'**

Look at primary controller logs storage209 logs and var/log/messages

**./log\_collector\_trigger.py -t all -F <from date> -T <to date> -n <name of primary ctlr> –n <name of storage node> –C**

Example:

**./log\_collector\_trigger.py -t all -F 2015-08-20 -T 2015-08-29 -n controller01 –n storage209 –C**

**LATTUS-M / [storage209] disk needs to be decommissioned**

Apparently needs to be manually decommissioned. Smart test recommended.

## BDY015 - How to manually decommission a disk.

**Introduction**

This KB article describes how to manually decommission a disk in an AmpliStor pool.

**Description**

The AmpliStor monitoring processes should identify bad disks and mark them as being degraded, so the administrator can decommission them later.  If for some reason a bad disk is not properly being marked as degraded, this procedure can be used to decommission the disk manually.

**Precaution**

The procedure below requires a machine and disk name to be entered. Mistyping these values may permanently decommission the wrong disk so its advise to double check your input.

**Procedure**

The following procedure should be run in the qshell on the management controller.

Create a connection to the cloudapi

|  |
| --- |
| api = i.config.cloudApiConnection.find('main') |

Retrieve the guid of the machine where the disk to be decommissioned resides.  Replace <machine\_name> with the hostname of that machine.

|  |
| --- |
| mguid = api.machine.find(name='<machine\_name>')['result'][0] |

Retrieve the guid of the disk that needs to be decommissioned. Replace <sdX> with the name of the disk, i.e. 'sde'.

|  |
| --- |
| dguid = api.disk.list(machineguid=mguid, name='<sdX>')['result'][0]['guid'] |

Verify the disk is in fact the correct one.

|  |
| --- |
| api.disk.list(dguid) |

Decommission the disk

|  |
| --- |
| api.machine.decommission\_disks(mguid, [dguid]) |

# STORAGExxx is HALTED

**LATTUS-Machine storage150 is HALTED on environment 'Apple Computer INC**

Associated with memory management bug. There is a patch being tested for this. QUANTUM-1098

[**Bug 53703**](http://bo.quantum.com/bugzilla/show_bug.cgi?id=53703) - Lattus storage nodes going to HALTED status unexpectedly **QUANTUM-867**

General description/summary:

Lattus storage nodes going to HALTED status unexpectedly

Customer Viewed Regression:

Customer has seen multiple storage nodes go into HALTED state for unknown

reason.

Additional Information:

Application: starting

Agent: BRDNLCN01-mgmt (A0:36:9F:0E:2A:9A)

Severity: CRITICAL

Occurrences: 2

First occurrence: 2014-11-24 07:27:12

Last occurrence: 2014-11-24 07:38:00

Message: Machine storage121 is HALTED

Workarounds:

Manually restart the storage node from the power button on the server.

BIOS Investigation:

Arne reported that he has looked at logs from the failing storage nodes:

Storage Nodes 089 and 167 reported many ECC errors indicating a memory problem,

and recommends either running a memtest or replacing the memory module

Storage Node 92 reported power errors indicating that the thresholds may be too

low

Storage Node 121 indicated that the watchdog feature may be enabled causing the

hang. Recommend disabling watchdog on this node

All other failed storage nodes did not have any output from the SCI logs.

Recommend checking to ensure logging is enabled and checking BIOS settings

Storage Node 1, which has not failed, did also indicate ECC errors, but it was

noted that the memory has already been replaced an

### SCS030 - HowTo Configure An Automated Reboot Of A Hung/Frozen node

# Filesystem errors on disk

**Lattus/ Filesystem errors on disk /dev/sdg, partition sdg1 on environment 'Apple Computer INC - CX1421CKC00016'**

File system errors after disk came back up. Suggest replacing the associated drives.

This bug shows how to do a health check

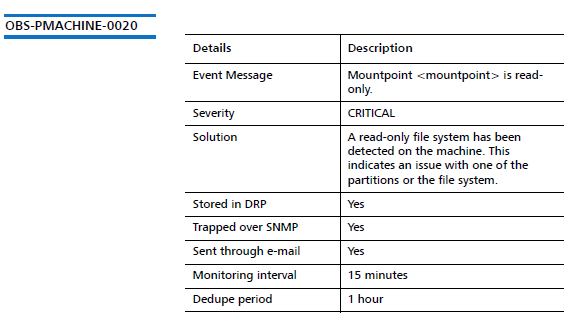
[**Bug 53451**](http://bo.quantum.com/bugzilla/show_bug.cgi?id=53451) - healthCheck aborts with varying errors on multiple runs

# Failure while executing event handling logic

Lattus / Failure while executing event handling logic for event type: **OBS-PMACHINE-0020** on environment 'Apple Computer INC - CX1421CKC00016'

Need to evaluate logs

Page 168 on Lattus Service Reference Guide



# Failed to execute monitoring rule

LATTUS-M/ Errors detected on machine storage175: Failed to execute monitoring rule check\_blockstores on environment 'Apple Computer INC - CX1421CKC00016'

Need to evaluate logs

[**Bug 45472**](http://bo.quantum.com/bugzilla/show_bug.cgi?id=45472) **- Amplidata Tracker:3.3.0 new install: Failed to execute monitoring function get\_disk\_info**

28-Oct Triage

- Waiting final analysis from AMP, but a release note would be useful to avoid

this false RAS event

- Workaround is to remove the USB stick before rebooting the node

- Cloning bug to request a release note

RAS Log: Failed to execute monitoring function get\_disk\_info

Resetting Cloud API Connection object, \* log: type:log time:2013/10/21

08:20:13 level:5 agent:0025905608D9.somewhere.com application:monitor\_agent

tags:

Created Cloud API Connection object Cloud API Connection

(<http://cloudapi:98d09d0e>-

[35e9-11e3-b4c6-0025905608d6@172.16.10.35](mailto:35e9-11e3-b4c6-0025905608d6@172.16.10.35):80/appserver/xmlrpc/), \* log:

type:log time:2013/10/21 08:20:13 level:6 agent:0025905608D9.somewhere.com

application:monitor\_agent tags:

Executing tasklet RemoveTracingAndLogging.py, \* log: type:log

time:2013/10/21 08:20:13 level:6 agent:0025905608D9.somewhere.com

application:monitor\_agent tags:

Executing tasklet FilterQShellErrors.py, \* log: type:log time:2013/10/21

08:20:13 level:6 agent:0025905608D9.somewhere.com application:monitor\_agent

[**Bug 54461**](http://bo.quantum.com/bugzilla/show_bug.cgi?id=54461) **- "Failed to execute monitoring rule check\_blockstores" event: Connection reset by peer**

I checked similar previous bugs/tickets and found a history of this error

associated with power failure, reboot, degraded disk, and bad SATA controller.

This time there is no apparent environmental issue. I am opening this

bug/ticket for further analysis. For reference here are related bugs/tickets:

BUG NUMBER - TICKET NUMBER

--------------------------

41878 - QUANTUM-293

41920 - QUANTUM-295

42315 - QUANTUM-326

45826 - QUANTUM-566

None - QUANTUM-606

# ALL ASSOCIATED WITH MEMORY MANAGEMENT BUG (QVC memory cleanup bug)

|  |
| --- |
| LATTUS/Swap usage is over 50% on environment 'Apple Computer INC - CX1421CKC00016'  Associated with memory management bug. There is a patch being tested for this. |
| LATTUS/System memory threshold (5MB) exceeded on environment 'Apple Computer INC - CX1421CKC00016'  Associated with memory management bug. There is a patch being tested for this.  **IF you are just getting these they are warnings and will work themselves down** |
| LATTUS/Load average over the last 15 minutes is high (41.74) on environment 'Apple Computer INC - CX1421CKC00016'  Associated with memory management bug. There is a patch being tested for this. |

QUANTUM-827

These events have been seen in Lattus 3.6.0

Bug 46906

Matt Taylor added a comment - Today 7:18 AM

Here are the steps used to disable swap usage monitoring on all storage nodes

Note: Best practice to verify the change on the first storage node prior to

updating all storage nodes. Edit the file and restart monitoring agent. If not

errors then proceed with all storage nodes.

SSH into first storage node: (or any storage node)

root@Controller1:~# ssh 10.10.1.41

Modify /opt/qbase3/apps/monitoring\_agent/rules.xml by commenting out the line

below.

root@Storage1:# vi /opt/qbase3/apps/monitoring\_agent/rules.xml

<get\_swap\_info>

<interval>900</interval>

# <!-- <rules>check\_swap\_usage</rules> -->

</get\_swap\_info>

Creating a cluster for all storage nodes of the environment (only running

nodes):

In []: ca = i.config.cloudApiConnection.find('main')

In []: minfo =

[(ca.machine.getManagementIpaddress(m['guid'],includevirtual=False)['result'],

ca.machine.getObject(m['guid']).accounts[0].password) for m in

ca.machine.list(machinerole='STORAGENODE')['result'] if m['status']=='RUNNING']

In []: minfo = zip(\*sorted(minfo, key=lambda x: int(x[0].split('.')[3])))

In []: ctrl\_clu = q.cluster.create(clustername='all\_storage\_nodes\_2',

domainname='tmp', ipaddresses=list(minfo[0]), masteripaddress=minfo[0][0],

superadminpassword=minfo[1][0], superadminpasswords=list(minfo[1][1:]))

In []: del ca,minfo

Saving the old version of file and replacing it with the updated version:

In []: strg\_clu = q.cluster.get('all\_storage\_nodes\_2')

In []: strg\_clu.execute('cp /opt/qbase3/apps/monitoring\_agent/rules.xml

/opt/qbase3/apps/monitoring\_agent/rules.orig' ,strg\_clu.listnodes())

In []: strg\_clu.sendfile('/opt/qbase3/apps/monitoring\_agent/rules.xml',

'/opt/qbase3/apps/monitoring\_agent/rules.xml', strg\_clu.listnodes())

Restarting the monitoring agent on all storage nodes to pick up the change:

In []: strg\_clu.executeQshell('q.manage.monitoringagent.restart()',

strg\_clu.listnodes())

# BOTH OF THESE WERE RELATED TO DISK REPLACEMENT

# Machine was rebooted

LATTUS/Machine was rebooted on environment 'Apple Computer INC - CX1421CKC00016'

Informational message

# New empty disk(s) detected

LATTUS/**New empty disk(s) detected** on environment 'Apple Computer INC - CX1421CKC00016'

Informational message

# Error retrieving repair statistics for daemon with id

These first two are due to logs rolling up if there is just one

monitor\_agent   storage039 (10:C3:7B:48:72:DE)  ERROR  1             2015-09-03 20:17:03       2015-09-03 20:17:03

Errors detected on machine storage039: Error retrieving repair statistics for daemon with id [99845693]

Errors detected on machine storage119: Error retrieving repair statistics for daemon with id [341836689]

Error retrieving repair statistics for daemon

Summary  
Python Library in all versions of Active Archive software and Amplidata/Lattus software will causes excessive alert events from the system due to leap year date handling

Event Type: OBS-APPLICATION-0050

Event Message: Errors detected on machine <node name>: Error retrieving repair statistics for daemon with id [<maintenance agent ID>]

Severity: ERROR

Problem Description:  
The issue surfaces due to a problem with the strptime library which does not properly handle the Feb 29th leap year date

Solution:  
This issue will resolve itself on March 1st and will not have any adverse effect on your storage IO operations.

# Errors detected on machine Error retrieving block store information

**This one I would need logs for**

**./log\_collector\_trigger.py -t all -F 2015-09-05 -T 2015-09-07 -n controller01 –n storage036 –C**

race condition between storage and maintenance agents

**monitoring agent : Error retrieving block store information for [0]**

Application          Agent    Severity                Occurrences       First occurrence                Last occurrence

monitor\_agent   storage036 (AC:22:0B:8A:4F:9C) ERROR 1             2015-09-06 12:17:02       2015-09-06 12:17:02

Errors detected on machine storage036: Error retrieving block store information for [423]

# Error retrieving storage daemon black lists

monitor\_agent s1a-storage006 (54:A0:50:87:11:9D) ERROR 1 2015-09-15 17:54:53 2015-09-15 17:54:53

Errors detected on machine s1a-storage006: Error retrieving storage daemon black lists for daemon with id [10]

CR41433 - monitoring agent will generate events that timeout when Apache is down due to log rolling and the message was triggered by apache restart. OK if occasional message.

# Storage pool is more than 70% full

workflowengine lmg-s1-c1 (A0:36:9F:61:DF:4E) WARNING 2 2015-09-16 08:59:48 2015-09-16 09:29:48

Storage pool is more than 70% full

An info message the customer had set the threshold to be notified

I have performed the following steps  
Edited this script: /opt/qbase3/apps/workflowengine/tasklets/rootobject/amplistor/aggregateStoragepoolInfo/amplistor\_aggregate\_storagepool\_info.py

1. elif used\_percentage > 80.0:
2. evt\_type\_id = 'AMPLISTOR-MON-DSS-STORAGEPOOL-0026'
3. threshold = 80
4. level = q.enumerators.ErrorconditionLevel.ERROR
5. elif used\_percentage > 70.0:
6. evt\_type\_id = 'AMPLISTOR-MON-DSS-STORAGEPOOL-0025'
7. threshold = 70
8. level = q.enumerators.ErrorconditionLevel.WARNING

Restart workflowengine via qshell:  
In [1]: q.manage.workflowengine.restart()  
Stopping the workflowengine.  
Stopped the workflowengine.  
Starting the workflowengine.  
Waiting for initialization  
Workflowengine started

We will watch the events over the next couple hours and see if they g

# Unverified objects

Application: monitor\_agent

Agent: controller01 (A0:36:9F:28:F9:D6)

Severity: CRITICAL

Occurrences: 1

First occurrence: 2015-09-21 07:01:26

Last occurrence: 2015-09-21 07:01:26

Message: Found [2725] unverified objects

It will eventually work them down if left alone.

If the events are a nuisance then there are ways to make the verification happen faster.  But it requires some manual work around and running of background verifications.

The goal of object verification is to make sure that all objects stored in Lattus are correctly stored and, in case of issues, that the object is repaired. The object verification process involves: • Checking whether the object exists • Checking whether all superblocks of the object exist: • For each superblock, check whether each blockstore contains enough checkblocks • For each checkblock, check whether the CRC32 is correct In Lattus, a default object verification process runs as a background process. The goal of this process is to verify all objects in a namespace within a certain time interval, which by default is one year.

Object verification is done in iterations because it is not possible to verify all objects at the beginning of the verification interval.

Objects that have the status “unverified”, have not been successfully verified in the proper verification interval

with the highest precautions because this action has a negative impact on Lattus performance due to temporarily placing a very high load on the system. Before forcing a namespace verification, you must estimate the minimum verification interval needed to verify all objects. SEE LATTUS SERVICE MANUAL page 305

# ZUFFA SR3582806 FYI storage011 issues

Andres storage011 was completely swapped out and ever since they have been having problems

Amplidata needs to look at the core logs

Have Jeff Quinn do the following (cores were done on 21st /filesystem errors were on 28th so the below brackets all of that –n controllers gets all 4 controllers)

**./log\_collector\_trigger.py -t all -F 2015-08-20 -T 2015-08-29 -n controllers –n storage011 –**

**Also from Storage011 get /var/log/messages that cover the above time period**

**And DSET from storage011**

Re: SR3572714 - ZUFFA, LLC dba ULTIMATE FIGHTING CHAMPIONSHIP (UFC) - LATTUS-M/storage011 having issues Warning Event Type OBS-GENERIC-0071

Output.

**root@storage011**:~# dmidecode | grep Vendo

Vendor: American Megatrends Inc.

root@storage011:~#  smartctl -x /dev/sda | grep Capacity

User Capacity:    4,000,787,030,016 bytes [4.00 TB]

**Storage011 (new one) is an S20 AS48 Aaeon**

LATTUS/: Event report 'CX1421CKC00012' - 'Zuffa LLC dba UFC'

Event report for environment CX1421CKC00012 - Zuffa LLC dba UFC -- 2015-08-28 16:52:19

Software version

3.6.0.27

Machines

Type Offline Decommissioned Total

Controller node 0 0 4

Storage node 0 0 130

Disks

Degraded Decomissioning Decommissioned Total

1 0 0 1576

MetaStores

Name Allocation type Degraded Status

env\_metastore SYSTEM No ACTIVE

framework SYSTEM No ACTIVE

data01 AUTO No ACTIVE

Storage pool

Worst case overall disk safety Used capacity Total capacity

4 3653.492 TiB (65.954%) 5539.464 TiB

Live events

Application Agent Severity Occurrences First occurrence Last occurrence

monitor\_agent storage011 (AC:22:0B:8A:AD:8C) CRITICAL 1 2015-08-28 16:00:06 2015-08-28 16:00:06

**Filesystem errors on disk /dev/sdj, partition sdj1**

monitor\_agent storage011 (AC:22:0B:8A:AD:8C) WARNING 1 2015-08-28 15:30:57 2015-08-28 15:30:57

**Core dump files found in /var/crash/ Filename: core\_storage011\_dss.bin\_**18651.1440179492.2015\_08\_21\_1440181021.gz (gzip compressed data, from Unix, last modified -- **Fri Aug 21 10:51:33 2015**)

monitor\_agent storage011 (AC:22:0B:8A:AD:8C) WARNING 1 2015-08-27 15:30:33 2015-08-27 15:30:33

Core dump files found in /var/crash/ Filename: core\_storage011\_dss.bin\_18651.1440179492.2015\_08\_21\_1440181021.gz (gzip compressed data, from Unix, last modified -- Fri Aug 21 10:51:33 2015)

All events since last report

Application Agent Severity Occurrences First occurrence Last occurrence

monitor\_agent storage011 (AC:22:0B:8A:AD:8C) CRITICAL 1 2015-08-28 16:00:06 **2015-08-28 16:00:06**

Filesystem errors on disk /dev/sdj, partition sdj1

osis controller01 (A0:36:9F:35:F3:DA) ERROR 1 2015-08-28 04:17:30 2015-08-28 04:17:30

Policy 'Aggregate Storagepool Info' failed on machines ['controller01']

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# UPGRADE ISSUE

[**Bug 57562**](http://bo.quantum.com/bugzilla/show_bug.cgi?id=57562) - Request research into IPMI, MEI drivers and Supermicro motherboards

QUANTUM-1124, HARDWARE-342

|  |  |
| --- | --- |
| **Version**: | 3.6.0 |

As a follow-up to the issues seen at Zuffa/UFC where the storage nodes do not

come up after a reboot during the upgrade from 3.5.1 to 3.6, we would like QTM

and Amplidata engineering to investigate workarounds and long term corrective

actions for this issue. A workaround was put into place at UFC to disable the

MEI drivers in Ubuntu. This may have a negative impact on the IPMI

functionality. The Apple Marcom upgrade was put on hold pending this

investigation, as well as direction for future upgrades. It has been reported

that we have not seen this issue on the Aaeon nodes as they do not use the

SuperMicro Motherboard, but would also like to confirm that the issue does not

impact the Aaeon/Asus MB.

CORRECTION: the S10 (Aeaon), S20 V1 (Aeaon) & S20 V2 (Sanmina) are susceptible

to this issue, and we have not seen it on the S30 (Sanmina).

We unpacked the storage nodes and powered them up. Both exhibit the “MEI

problem” (see attached IMG\_9860.jpg). Both of the storage nodes successfully

PXE booted after we adjusted their BIOS settings to PXE instead of booting off

hard drive. Lattus (we’re running 3.6.0.27) was able to see the nodes.

# Error retrieving storage daemon info for daemon with id

monitor\_agent s1a-storage009 (60:A4:4C:E7:1B:3C)         ERROR  1              2015-09-08 17:16:01        2015-09-08 17:16:01

Errors detected on machine s1a-storage009: Error retrieving storage daemon info for daemon with id [16]

That means that when the monitoring agent attempted to collect information on s1a-storage009 it was unable to get the information about storage daemon 16.

Are there other events at the time for s1a-storage009?

Is the storagedaemon running on s1a-storage009? In [1]: q.dss.storagedaemons.getStatus()

Are there other events at the time for other storage nodes?

Is the monitoring agent running on the controller node? In [53]: q.manage.monitoringagent.getStatus()