



# **coredump events**

## **ONTAP EMS reference**

NetApp  
November 22, 2024

# Table of Contents

- coredump events ..... 1
- coredump.all events ..... 1
- coredump.compress events ..... 1
- coredump.compression events ..... 1
- coredump.coreid events ..... 2
- coredump.delete events ..... 2
- coredump.disk events ..... 3
- coredump.dr events ..... 3
- coredump.dump events ..... 5
- coredump.encryption events ..... 12
- coredump.findcore events ..... 13
- coredump.host events ..... 15
- coredump.job events ..... 16
- coredump.long events ..... 17
- coredump.mcc events ..... 18
- coredump.micro events ..... 18
- coredump.nosavecore events ..... 19
- coredump.nvlog events ..... 19
- coredump.save events ..... 20
- coredump.segment events ..... 25
- coredump.shutdown events ..... 26
- coredump.spare events ..... 26
- coredump.sparecore events ..... 28
- coredump.sparecoreinit events ..... 29

# coredump events

## coredump.all events

### coredump.all.disks.used

#### Severity

ERROR

#### Description

All disks available to coredump already contain unsaved cores.

#### Corrective Action

There are three choices. First, add more disks to the system. Second, delete any unsaved cores by running "savecore -k". Third, save any unsaved cores by running "savecore". It may be necessary to run "savecore -f" to save the cores, if previous attempts to save these cores have failed.

#### Syslog Message

No disks available for dumpcore

#### Parameters

(None).

## coredump.compress events

### coredump.compress.failed

#### Severity

ERROR

#### Description

An error was encountered while compressing data for the coredump. This core will be deleted.

#### Corrective Action

(None).

#### Syslog Message

Error compressing core data: %s

#### Parameters

msg (STRING): The error message

## coredump.compression events

### coredump.compression.details

#### Severity

INFORMATIONAL

## Description

This message occurs when a compressed kernel core is generated. Details about compression are reported.

## Corrective Action

(None).

## Syslog Message

%s, compression time %llu msec, compression count %llu, bytes before compression %llu, bytes after compression %llu, compression ratio %llu%%, compression bandwidth %llu MB/s, total compression time %llu msec, all-zero chunk count %llu.

## Parameters

**type** (STRING): Type of dump. Possible types are compressed core, compressed spraycore, spraycore and sparecore.

**compressTime** (LONGINT): Compression time, in milliseconds.

**compressCount** (LONGINT): Number of compressions.

**compressBytesIn** (LONGINT): Number of bytes before compression.

**compressBytesOut** (LONGINT): Number of bytes after compression.

**compressRatio** (LONGINT): Compression ratio expressed as a percentage.

**compressBandwidth** (LONGINT): Compression bandwidth, in MBps.

**totalCompressTime** (LONGINT): Total compression time, in milliseconds.

**allZeroChunks** (LONGINT): Number of all-zero chunks.

# coredump.coreid events

## coredump.coreid.not.found

### Severity

INFORMATIONAL

### Description

Could not find the specified core

### Corrective Action

Double-check the core id, and try again.

### Syslog Message

Core %d could not be found

### Parameters

**coreid** (INT): The core id that could not be found

# coredump.delete events

## coredump.delete.denied

### Severity

INFORMATIONAL

## Description

This message occurs when a requested coredump delete cannot occur because an aggregate relocation or a takeover is in progress. The system cannot delete a core while either an aggregate relocation or a takeover is in progress.

## Corrective Action

Retry the coredump delete after aggregate relocation or takeover completes.

## Syslog Message

Coredump delete disallowed because an aggregate relocation or a takeover in progress.

## Parameters

(None).

# coredump.disk events

## coredump.disk.write.details

### Severity

INFORMATIONAL

### Description

This message occurs when a kernel core is written to disk. Details about disk operations are reported.

### Corrective Action

(None).

### Syslog Message

%s, disk write time %llu msec, disk write count %llu, disk write bytes %llu, disk write bandwidth %llu MB/s.

### Parameters

**type** (STRING): Type of dump. Possible types are compressed core, compressed spraycore, spraycore and sparecore.

**writeTime** (LONGINT): Disk write time, in milliseconds.

**writeCount** (LONGINT): Number of disk writes.

**writeBytes** (LONGINT): Number of bytes written to disk.

**writeBandwidth** (LONGINT): Disk write bandwidth, in MBps.

# coredump.dr events

## coredump.dr.dump.not.started

### Severity

ERROR

### Description

This message occurs when coredump subsystem detects that disaster recovery (DR) partner sparecore dump did not start.

### **Corrective Action**

Log in to service processor and power cycle the DR partner that may not have started dumping core.

### **Syslog Message**

DR partner sparecore dump failed to start.

### **Parameters**

(None).

## **coredump.dr.spare.abort**

### **Severity**

INFORMATIONAL

### **Description**

This message occurs when coredump subsystem attempts to abort any sparecore dump that might be in progress on disaster recovery (DR) partner. This process can take up to 60 seconds. During this time, other coredump operations are not serviced.

### **Corrective Action**

(None).

### **Syslog Message**

Pausing to abort DR partner sparecore dump on %s because %s.

### **Parameters**

**diskuid** (STRING): UID of DR partner sparecore dump disk.

**reason** (STRING): Reason for aborting the DR partner sparecore dump.

## **coredump.dr.spare.noAbort**

### **Severity**

ERROR

### **Description**

This message occurs when coredump subsystem is unable to abort any sparecore dump that might be in progress on disaster recovery (DR) partner.

### **Corrective Action**

Log in to service processor and power cycle the DR partner that is still dumping core.

### **Syslog Message**

DR partner sparecore dump could not be aborted on disk %s because %s.

### **Parameters**

**diskuid** (STRING): UID of DR partner sparecore dump disk.

**reason** (STRING): Reason for abort failure.

# coredump.dump events

## coredump.dump.abort

### Severity

ERROR

### Description

This message occurs when there is an unsaved core on the dump device. The abort prevents a new crash from overwriting the original dump core, which is expected to contain more interesting information for analysis.

### Corrective Action

If the default behavior is not desired, then set the boot option 'save-latest-core?' to true.

### Syslog Message

Dump is aborted because a core in the dump device is not saved yet.

### Parameters

(None).

## coredump.dump.disk.failed

### Severity

NOTICE

### Description

This message occurs once for each core disk that has read or write errors while dumping the core. Each disk is marked as failed, so further writes to the disks are dropped. Dumpcore tries to recover from this error, and continue dumping the core.

### Corrective Action

(None).

### Syslog Message

Disk %s had %u read and %u write error(s) during dumpcore.

### Parameters

**disk** (STRING): Name of this core disk marked failed with read or write errors.

**readErrorCount** (INT): Number of read errors seen on this core disk.

**writeErrorCount** (INT): Number of write errors seen on this core disk.

## coredump.dump.disk\_failed.count

### Severity

ERROR

### Description

During the last dumpcore attempt, disk write errors are ignored, and dumpcore continues. In this case, too many disks failed during the last dump attempt for savecore to create a core file. The core will be deleted, and no core will be available from this panic.

## Corrective Action

(None).

## Syslog Message

%d of %d disks failed during dumpcore

## Parameters

**failed** (INT): Number of failed disks

**total** (INT): Number of disks used

## coredump.dump.disks.count

## Severity

ERROR

## Description

This message occurs when the system fails to write a coredump due to lack of disk space.

## Corrective Action

Disk space needed for a coredump can be reduced by enabling sparse cores, which is the default setting. Verify that sparse cores are enabled by using the "system node coredump config show" command. If not, enable them by using the "system node coredump modify -node local -sparse-enabled true" command. If there is still not enough disk space to write a coredump, add a spare disk that is at least 0.6 times the sum of primary memory size and NVRAM memory size. You can obtain sizes of primary memory and NVRAM memory by using the "run local sysconfig" command and reading entries for system board and NVRAM.

## Syslog Message

Not enough disks to write core dump.

## Parameters

(None).

## coredump.dump.dont

## Severity

INFORMATIONAL

## Description

This message occurs when a core dump is not performed because the system was configured to not dump cores or dump is disabled in the code path that caused the system to panic. Any of the following configuration methods can disable coredump: - Environment variable "dont-dump-core?" is "true" - Parameter "-coredump-attempts" in the command "system node coredump config" is "0".

## Corrective Action

If the reason is "dump disabled in the current code path", there is no corrective action that can be taken. If the reason is "coredump configured to not dump cores", 1. "-coredump-attempts" parameter might be set to "0". Check the "Max Dump Attempts" value in the output of the "system node coredump config show -node node\_name" command. If "0", use the "system node coredump config modify ←node node\_name> -coredump-attempts <number\_of\_attempts>" command to set a non-zero value. The default is 2 attempts. 2. "dont-dump-core?" environment variable might be set to "true". From the nodeshell, use the (priv: diag) "bootargs dump" command to check whether "dont-dump-core?" exists and is set to "true". To unset the variable, use the "bootargs unset dont-dump-core?" command.



**Syslog Message**

Coredump not performed because %s.

**Parameters**

**reason** (STRING): Reason that coredump was not performed.

**coredump.dump.failed****Severity**

EMERGENCY

**Description**

Coredump failed to complete.

**Corrective Action**

Previously logged coredump.dump.\* events should help explain the exact cause of the failure.

**Syslog Message**

Dumpcore failed

**Parameters**

(None).

**coredump.dump.find.all.used****Severity**

ERROR

**Description**

Unsaved cores occupy all the disks available to dumpcore.

**Corrective Action**

Save the unsaved cores using "savecore" or "savecore -f". If the cores aren't needed, run "savecore -k" to delete them.

**Syslog Message**

All available disks contain unsaved cores

**Parameters**

(None).

**coredump.dump.find.disks.failed****Severity**

ERROR

**Description**

Could not find any suitable disks for dumping a core.

**Corrective Action**

Make sure there are non-broken disks connected to (and owned by) the panicing host.

**Syslog Message**

No valid disks found

**Parameters**

(None).

**coredump.dump.find.spare.failed****Severity**

ERROR

**Description**

A sparecore dump is supposed to be done, but no suitable spare disk can be found. In these situations, it is more important to allow a fast takeover than it is to get the core, so the dump is terminated, and takeover begins.

**Corrective Action**

Add a spare disk to the system, or disable the "cf.takeover.on\_panic" option.

**Syslog Message**

No valid spare disk found

**Parameters**

(None).

**coredump.dump.nvram.copy.failed****Severity**

ERROR

**Description**

Error while copying NVRAM into main memory during dumpcore. The resulting coredump will have dummy data in the NVRAM segment.

**Corrective Action**

(None).

**Syslog Message**

Error while copying NVRAM contents into main memory

**Parameters**

(None).

**coredump.dump.raid.notready****Severity**

ERROR

**Description**

RAID has not yet processed the labels on the disks, so dumpcore cannot proceed. This is typically the case for panics very early during boot.

**Corrective Action**

(None).

**Syslog Message**

RAID not ready to dump cores

**Parameters**

(None).

**coredump.dump.recursive.late****Severity**

ERROR

**Description**

A recursive panic situation has been encountered. The location of the second panic makes it unsafe to attempt another dump.

**Corrective Action**

(None).

**Syslog Message**

Late recursive panic caused dumpcore to abort

**Parameters**

(None).

**coredump.dump.save.latest.core****Severity**

ERROR

**Description**

The boot option `save-latest-core?` is set to true. This will discard all unsaved cores during coredump.

**Corrective Action**

If the behavior is not desired then set the boot option `save-latest-core?` to false.

**Syslog Message**

Saving the latest core and discarding all unsaved cores.

**Parameters**

(None).

**coredump.dump.scrub.skipped**

**Severity**

NOTICE

**Description**

This message occurs when the coredump process was asked to scrub sensitive memory areas but could not do so. The dump must be aborted to avoid saving such memory areas to disk.

**Corrective Action**

(None).

**Syslog Message**

Could not scrub protected memory regions.

**Parameters**

(None).

**coredump.dump.segments.count****Severity**

ERROR

**Description**

Dumpcore ran out of space for the core segments in the header. This core cannot be dumped.

**Corrective Action**

Disable the "coredump.metadata\_only" option.

**Syslog Message**

Out of segment space in the core header

**Parameters**

(None).

**coredump.dump.spare.no.space****Severity**

ERROR

**Description**

The selected spare disk might not be big enough to hold the core. The dump attempt has been aborted.

**Corrective Action**

Make sure a spare disk is available that is at least as big as the sum of main memory and NVRAM. Set the option "coredump.metadata\_only" to "on". If the problem persists, turn off the option "cf.takeover.on\_panic". This will disable the sparecore feature, and delay takeovers until the panicing partner has finished dumping the core.

**Syslog Message**

Spare disk too small for coredump

**Parameters**

(None).

**coredump.dump.started****Severity**

INFORMATIONAL

**Description**

Basic info about start of dumpcore

**Corrective Action**

(None).

**Syslog Message**

%s starting with %d disks

**Parameters**

**type** (STRING): Type of dump. Possible types are compressed, sprayed, and sparecore.

**disks** (INT): Number of disks being used

**coredump.dump.time****Severity**

INFORMATIONAL

**Description**

Information about how long dumpcore took to complete.

**Corrective Action**

(None).

**Syslog Message**

%s completed in %d seconds

**Parameters**

**type** (STRING): Type of dump. Possible types are compressed, sprayed, and sparecore.

**seconds** (INT): Number of seconds spent on the dump

**coredump.dump.write.blocks.failed****Severity**

ERROR

**Description**

Failed to write dump data blocks to disk. If the write failed because of disk problems, dumpcore will restart, without the disk that failed. If the write failed for other reasons (eg. attempt to write to a bad position on the disk), dumpcore will be aborted.

### **Corrective Action**

(None).

### **Syslog Message**

Error dumping %llu blocks starting at block %llu to disk %s

### **Parameters**

**blocks** (LONGINT): Number of blocks trying to be dumped

**start** (LONGINT): Starting block number

**disk** (STRING): Name of disk on which the write failed

## **coredump.dump.write.error**

### **Severity**

ERROR

### **Description**

An i/o error occurred while attempting to write a coredump to disk.

### **Corrective Action**

(None).

### **Syslog Message**

An i/o error occurred while attempting to dump core.

### **Parameters**

(None).

## **coredump.dump.write.header.failed**

### **Severity**

ERROR

### **Description**

Failed to write the core header to disk during dumpcore.

### **Corrective Action**

(None).

### **Syslog Message**

Error writing core header to block %llu on disk %s

### **Parameters**

**header** (LONGINT): Block number of the core header block

**disk** (STRING): Name of disk on which the write failed

## **coredump.encrypted events**

## **coredump.encryption.disabled**

### **Severity**

ERROR

### **Description**

This message occurs when encryption is configured on the controller, but the coredump subsystem cannot encrypt data. A system panic in this situation results in no core file being produced.

### **Corrective Action**

(Call support).

### **Syslog Message**

Coredump encryption cannot encrypt data. Core file will not be saved in this state.

### **Parameters**

(None).

## **coredump.encryption.restored**

### **Severity**

NOTICE

### **Description**

This message occurs when encryption is configured on the controller, the coredump subsystem experienced a transient problem encrypting data, but has since recovered and will be able to save encrypted core dump files in the event of a system panic.

### **Corrective Action**

(None).

### **Syslog Message**

Coredump encryption experienced a failure in the past but has recovered. Core files can now be saved.

### **Parameters**

(None).

## **coredump.findcore events**

### **coredump.findcore.error**

### **Severity**

ERROR

### **Description**

This message occurs when the core dump process detects an error while checking for an unsaved core in the dump device. The dump is not aborted.

### **Corrective Action**

(None).

## Syslog Message

Ignored the error (%d: %s) when trying to find an unsaved core in the dump device.

## Parameters

**errCode** (INT): Error code.

**errString** (STRING): Error code represented as a string.

## coredump.findcore.nospace

### Severity

ERROR

### Description

There are too many unsaved cores on the system to handle them all at one time. Some cores are being ignored.

### Corrective Action

Run "savecore" multiple times to save all the cores. Alternatively, run "savecore -k" to delete all of them.

## Syslog Message

Too many unsaved cores

## Parameters

(None).

## coredump.findcore.partial

### Severity

ERROR

### Description

This message occurs during a core dump, when only a partial core is found. Although enough of this core is present for the core to be partially saved, there might be errors while doing so. These occur when data essential to the core creation process is lost with the missing disks.

### Corrective Action

Run the "savecore" command to attempt to save the partial core, or run the "savecore -k" command to delete all unsaved cores.

## Syslog Message

Partial core %s is missing %d of %d disks.

## Parameters

**corefile** (STRING): Name of the kernel core file.

**missing** (INT): Number of disks that are missing.

**total** (INT): Total number of disks used in this dump.

## coredump.findcore.partial.nosave



**Severity**

ERROR

**Description**

This message occurs when a partial core is found with too much missing to be saved.

**Corrective Action**

Run the "savecore -k" command to delete all unsaved cores.

**Syslog Message**

Cannot save partial core %s, missing %d of %d disks.

**Parameters**

**corefile** (STRING): Name of the kernel core file.

**missing** (INT): Number of disks that are missing.

**total** (INT): Total number of disks used in this dump.

## coredump.host events

### coredump.host.spare.none

**Severity**

INFORMATIONAL

**Description**

This message occurs when no sparecore disk is found at boot/takeover time.

**Corrective Action**

(None).

**Syslog Message**

No sparecore disk was found for host %d.

**Parameters**

**host** (INT): Host for which no sparecore disk was found.

### coredump.host.spare.save

**Severity**

INFORMATIONAL

**Description**

This message occurs to tell you which disk was identified as the sparecore disk at boot/takeover time.

**Corrective Action**

(None).

**Syslog Message**

Disk %s was identified as the sparecore disk at boot/takeover time for host %d.

## Parameters

**diskname** (STRING): Name of the sparecore disk.  
**host** (INT): Host that the sparecore disk belongs to.

## coredump.host.spare.set

### Severity

INFORMATIONAL

### Description

This message occurs when a disk is identified as a sparecore disk and the host's sparecoreinfo is updated during assimilation.

### Corrective Action

(None).

### Syslog Message

(None).

## Parameters

**disk** (STRING): Name of the sparecore disk  
**host** (INT): Host whose sparecoreinfo was updated.

## coredump.job events

### coredump.job.notstarted

#### Severity

ERROR

#### Description

This message occurs when the system can not automatically save a core failed.

#### Corrective Action

Use the 'system node coredump save' command to save the core.

#### Syslog Message

(None).

#### Parameters

(None).

### coredump.job.notswitchedback

#### Severity

ERROR

#### Description

This message occurs when the system cannot automatically save cores because a MetroCluster(tm) switchback was not performed.

### Corrective Action

Return storage to the MetroCluster member by using the "metrocluster switchback" command, and then use the "coredump save-all" command to save any unsaved cores.

### Syslog Message

System cannot automatically save cores because a MetroCluster switchback was not performed.

### Parameters

(None).

## coredump.long events

### coredump.long.queued.msg

#### Severity

NOTICE

#### Description

This message occurs when the coredump manager thread detects that a message has been queued longer than expected, as defined by an internal threshold. System features and performance are not affected.

#### Corrective Action

(None).

#### Syslog Message

Coredump manager starts processing a long queued message: "%s" queued "%lu" ms.

#### Parameters

**message\_type** (STRING): Coredump message type.  
**queue\_time** (LONGINT): Message queued duration, in milliseconds.  
**threshold** (LONGINT): Message queued threshold, in milliseconds.

### coredump.long.running.msg

#### Severity

NOTICE

#### Description

This message occurs when the coredump manager thread has finished processing a message that took longer than expected, as defined by an internal threshold.

#### Corrective Action

(None).

#### Syslog Message

Coredump manager finished processing a long running message: %s took %lu ms.

#### Parameters

**message\_type** (STRING): Coredump message type.  
**duration** (LONGINT): Message processing duration, in milliseconds.  
**threshold** (LONGINT): Message processing threshold, in milliseconds.

# coredump.mcc events

## coredump.mcc.nso.veto

### Severity

INFORMATIONAL

### Description

This message occurs when the coredump subsystem vetoes MetroCluster(tm) negotiated switchover because a coredump save is in progress.

### Corrective Action

Use the "node coredump status" command to verify the completion of the active coredump save, or use the "-override-vetoes true" option when performing MetroCluster switchover. Using the "-override-vetoes true" option will abort the active coredump save.

### Syslog Message

MetroCluster switchover vetoed because a coredump save was in progress.

### Parameters

(None).

# coredump.micro events

## coredump.micro.completed

### Severity

INFORMATIONAL

### Description

Microcore generation has completed.

### Corrective Action

(None).

### Syslog Message

Microcore (%s) generation completed

### Parameters

**file** (STRING): Pathname of microcore that was created

## coredump.micro.failed

### Severity

ERROR

### Description

Microcore generation has failed.

**Corrective Action**

(Call support).

**Syslog Message**

Generation of microcore failed

**Parameters**

(None).

## **coredump.nosavecore events**

### **coredump.nosavecore.noflush**

**Severity**

NOTICE

**Description**

This message occurs when the process that saves a coredump is not able to flush to the boot device at the end of coredump.

**Corrective Action**

(None).

**Syslog Message**

Coredump was not able to flush the remaining data to the boot device. Error: %llu.

**Parameters**

`dumpcore_error` (LONGINTEX): Coredump error code.

## **coredump.nvlog events**

### **coredump.nvlog.replay.dump.dont**

**Severity**

NOTICE

**Description**

This message occurs when one or more unsuccessful attempts have been made to process information in the WAFL® NVlog after a previous failure. The number of allowed core dumps has been exceeded and this core dump will not be taken.

**Corrective Action**

The node will reboot. No further action is necessary.

**Syslog Message**

Core dump was not taken for the latest failure while processing the WAFL NVlog.

**Parameters**

(None).

# coredump.save events

## coredump.save.aborted

### Severity

NOTICE

### Description

This message occurs when the savecore operation is aborted while saving a kernel core. The core is preserved for a future iteration of savecore.

### Corrective Action

(None).

### Syslog Message

%s processing savecore aborted%s%s.

### Parameters

**file** (STRING): Name of the kernel core file being created.

**operation** (STRING): Type of operation that is being attempted when the abort occurs.

**action** (STRING): Type of action being taken when the abort occurs.

## coredump.save.attempts.count

### Severity

NOTICE

### Description

Previous attempts to save this core have failed, either due to reboots or panics. To prevent a panic loop, savecore is aborting this attempt.

### Corrective Action

Use either "savecore -f" to save the core, or "savecore -k" to erase the core.

### Syslog Message

Too many attempts to save this core

### Parameters

(None).

## coredump.save.completed

### Severity

NOTICE

### Description

This message occurs when all the core segments belonging to a coredump have been saved.

### Corrective Action

(None).

## Syslog Message

Saved core %s for node %s in %llu seconds. Core size is %llu bytes.

## Parameters

**file** (STRING): Primary core segment file that was created.

**node** (STRING): Name of the node whose core was saved.

**seconds** (LONGINT): Number of seconds spent on saving all the core segments.

**size** (LONGINT): Size of primary core segment in bytes.

**chksum** (STRING): Checksum of primary core segment file's data.

## coredump.save.denied

### Severity

INFORMATIONAL

### Description

This message occurs when the coredump subsystem cannot finish saving a core because an aggregate relocation or a takeover is in progress. The system cannot save a core while either an aggregate relocation or a takeover is in progress.

### Corrective Action

(None).

## Syslog Message

Coredump save disallowed because an aggregate relocation or a takeover in progress.

## Parameters

(None).

## coredump.save.disk.missing

### Severity

INFORMATIONAL

### Description

A disk being used by coredump is now missing. This could be due to the ownership of the disk changing, the disk being removed, etc. Most savecore operations will proceed, but if further errors are encountered, or an expected core cannot be found, corrective action will be necessary.

### Corrective Action

If this disk has failed, no action is possible. If this message was seen while running a "savecore -s", "savecore -i", or "savecore -l" operation, replace the missing disk. This might be done by undoing the ownership change, or reinserting the missing disk. If this message was seen while saving the core, and no further errors were seen, no action is necessary. If more errors were seen, replace the missing disk as above, and run "savecore" again.

## Syslog Message

Disk %s being used by coredump is missing

## Parameters

**disk** (STRING): Name of the missing disk

## **coredump.save.diskio.zone**

### **Severity**

ERROR

### **Description**

A disk IO operation was attempted outside of the allowed regions on the disk. This behavior indicates a corrupt core, and it will be deleted.

### **Corrective Action**

(None).

### **Syslog Message**

IO to %d block(s) starting at %llu on disk %s is not allowed

### **Parameters**

**count** (INT): Number of blocks in IO request

**start** (LONGINT): Starting block number on the disk

**disk** (STRING): Name of the disk causing trouble

## **coredump.save.error**

### **Severity**

NOTICE

### **Description**

A correctable error was encountered while saving the core. Savecore will continue saving other cores.

### **Corrective Action**

Once savecore finishes, address errors that may have appeared, and restart savecore.

### **Syslog Message**

%s processing encountered error

### **Parameters**

**file** (STRING): The core file that savecore was attempting to create

## **coredump.save.failed**

### **Severity**

NOTICE

### **Description**

An unrecoverable error was encountered while saving the core. This core will be deleted, and savecore will continue saving other cores.

### **Corrective Action**

(None).

### **Syslog Message**

%s processing failed



## Parameters

**file** (STRING): The core file that savecore was attempting to create

## coredump.save.internal.error

### Severity

ERROR

### Description

An internal coredump error was encountered. This indicates the dumped core is corrupt, and will be deleted.

### Corrective Action

(None).

### Syslog Message

Internal coredump error

### Parameters

(None).

## coredump.save.nospace

### Severity

ALERT

### Description

This message occurs when there is not enough space in the root volume to save the core. Savecore is making a worst-case estimate when determining how much space might be needed to save the core, and does not start unless that much space is available.

### Corrective Action

Try performing the following steps to make sufficient room in root volume: Delete old or unneeded core files by using the "coredump delete" command. Delete old or unneeded core file segments by using the "coredump segment delete" command. Delete any Snapshot@ copies that include the deleted files. Adjust minimum free space on the root file system by using "coredump config modify" command. Once there is sufficient room, take the following steps to save the cores: To save all core files using "coredump save-all" command, run the "coredump status -instance" command and ensure that "Space Available On Internal Filesystem" is greater than "Space Needed to Save All Unsaved Cores" plus "Minimum Free Bytes on Root Filesystem". To save a specific core file, run the "coredump save" command, when available space of "Space Needed To Save Core" (from the "coredump show" command) plus "Minimum Free Bytes on Root Filesystem" is sufficient.

### Syslog Message

Available space (%llu %dKB blocks) is not enough to save the core image (requires up to %llu blocks) and maintain the minimum free space of %llu blocks.

### Parameters

**avail** (LONGINT): Number of blocks available for saving the core image.

**blocksize** (INT): Block size in kilobytes.

**needed** (LONGINT): Number of blocks needed for saving the core image.

**minfree** (LONGINT): Number of blocks reserved for normal system operation.

## **coredump.save.partial.not.ok**

### **Severity**

INFORMATIONAL

### **Description**

Can not save partial core that was dumped using disks with different sizes. Savecore can not determine the size of the missing disks. Attempting a save in this condition will most likely end up in an error.

### **Corrective Action**

Run "savecore -k" to delete all unsaved cores.

### **Syslog Message**

Cannot save partial core (%s) that used disks with different sizes, missing %llu of %llu disks

### **Parameters**

**panic** (STRING): Panic string of the partial core.

**missing** (LONGINT): The number of disks that are missing

**total** (LONGINT): Total number of disks used in this dump

## **coredump.save.read.header.failed**

### **Severity**

NOTICE

### **Description**

An error was encountered while trying to read the core header block from disk.

### **Corrective Action**

(None).

### **Syslog Message**

Could not read core header block (%llu) from disk %s

### **Parameters**

**block** (LONGINT): Block number being searched for the core header

**disk** (STRING): Name of the disk that could not be read

## **coredump.save.started**

### **Severity**

INFORMATIONAL

### **Description**

This message occurs when the savecore operation starts saving a kernel core.

### **Corrective Action**

(None).

### **Syslog Message**

Saving %llu MB to %s via %s ("%s") for node %s.

## Parameters

**size** (LONGINT): Number of MB being saved.  
**file** (STRING): Name of the kernel core file to be created.  
**dumpcoretype** (STRING): Kernel coredump type.  
**panic** (STRING): Panic string of the kernel core being saved.  
**node** (STRING): Name of the node whose core is being saved.

# coredump.segment.events

## coredump.segment.auto.fail

### Severity

NOTICE

### Description

This message occurs when the system encounters an error while trying to send a core segment request.

### Corrective Action

(None).

### Syslog Message

Failed to start automatic segmenting for core file, %s , for host %d.

## Parameters

**file** (STRING): Name of the core file that would have been segmented.  
**host** (INT): Host that owns the core file.

## coredump.segment.auto.pfail

### Severity

INFORMATIONAL

### Description

This message occurs when the system cannot start the segmenting of a core file because the node is no longer in takeover mode.

### Corrective Action

(None).

### Syslog Message

Cannot start segmenting a core file, %s. Node is no longer in takeover mode.

## Parameters

**file** (STRING): Name of core file that would have been segmented.

## coredump.segment.completed

### Severity

NOTICE

## Description

This message occurs when the segmenting of a full core file has been completed.

## Corrective Action

(None).

## Syslog Message

Saved core segment %s for node %s. Core size is %llu bytes and core data checksum is %s.

## Parameters

**file** (STRING): Name of the core file segmented.

**node** (STRING): Name of the node whose core segment was saved.

**size** (LONGINT): Size of core segment in bytes.

**chksum** (STRING): Checksum of the core segment file's data.

# coredump.shutdown events

## coredump.shutdown.trace

### Severity

NOTICE

### Description

This message occurs when a core is about to be generated. Durations of various activities at that time are reported.

### Corrective Action

(None).

### Syslog Message

coredump shutdown times: kmod dumper start %llu, panic\_info set %llu, sf\_dumpcore sent %llu, sparecore id sent %llu, fmot\_outage info sent %llu, disk\_dump\_start done %llu, coredump begin %llu (all in msec).

### Parameters

**kmodDumperStartTime** (LONGINT): Duration from panic start to dump processing start, in milliseconds.

**panicInfoTime** (LONGINT): Duration until panic information is collected, in milliseconds.

**sfDumpcoreSentTime** (LONGINT): Duration until SF\_DUMPCORE state is sent over the interconnect, in milliseconds.

**sparecoreSentTime** (LONGINT): Duration for sending the sparecore ID over the interconnect, in milliseconds.

**fmotOutageSentTime** (LONGINT): Duration for sending FMOT (Failover Monitor Outage Tracker) Outage information over the interconnect, in milliseconds.

**diskDumpStartTime** (LONGINT): Duration until disk dump start stage, in milliseconds.

**coredumpBeginTime** (LONGINT): Duration until the start of the actual core dump, in milliseconds.

# coredump.spare events

## coredump.spare.abort

**Severity**

INFORMATIONAL

**Description**

This message occurs when coredump attempts to terminate any sparecore dump that might be in progress on the partner. This process can take up to 60 seconds. During this time, other coredump operations are not serviced.

**Corrective Action**

(None).

**Syslog Message**

Pausing to abort partner sparecore dump because %s.

**Parameters**

**reason** (STRING): Reason for aborting the partner sparecore dump.

**coredump.spare.abort.failed****Severity**

ERROR

**Description**

Unable to reserve the sparecore disk. This disk cannot be used by RAID until it is successfully reserved.

**Corrective Action**

If the other partner can be booted, do a giveback. Otherwise, periodic attempts will automatically be made to reserve the disk in question.

**Syslog Message**

Sparecore disk could not be reserved

**Parameters**

(None).

**coredump.spare.abort.release****Severity**

NOTICE

**Description**

This message occurs when the system encounters an error while trying to release the disk reservation from a disk.

**Corrective Action**

(None).

**Syslog Message**

Could not release disk reservation from disk %s (0x%x)

## Parameters

**diskname** (STRING): Name of the disk whose disk reservation could not be released.

**diskid** (INHEX): ID of the disk whose disk reservation could not be released.

## coredump.spare.dump

### Severity

INFORMATIONAL

### Description

This message occurs to tell you which disk was used as the sparecore disk at dumpcore time.

### Corrective Action

(None).

### Syslog Message

Disk %s was used as the sparecore disk at dumpcore time.

## Parameters

**diskname** (STRING): Name of the sparecore disk.

## coredump.spare.invalid.size

### Severity

ERROR

### Description

This message occurs when coredump subsystem encounters a spare disk with an invalid size. If an unsaved core file exists on this disk, it will be deleted because it can no longer be saved.

### Corrective Action

Run the "storage disk show -disk disk\_name" command to make sure that the disk is still connected as a spare disk. If so, replace the disk. If not, the disk is part of an aggregate or is in the process of being assigned to one, and no action is required.

### Syslog Message

sparecore disk: %s has invalid size: %llu, skipping it from all coredump subsystem operations

## Parameters

**disk** (STRING): Name of the disk.

**size** (LONGINT): Size available on the disk, reported by RAID.

## coredump.sparecore events

### coredump.sparecore.killed

### Severity

INFORMATIONAL

**Description**

Sparecore was aborted by partner.

**Corrective Action**

(None).

**Syslog Message**

Sparecore was aborted by partner.

**Parameters**

(None).

## **coredump.sparecoreinit events**

### **coredump.sparecoreinit.times**

**Severity**

NOTICE

**Description**

This message occurs when sparecore initialization is completed. Durations of constituent activities within sparecore initialization are reported.

**Corrective Action**

(None).

**Syslog Message**

sparecore\_init: update firmware %llu msec, write partial headers %llu msec, clear abort area %llu msec, write node status %lld msec.

**Parameters**

**fwUpdateTime** (LONGINT): Time taken to update firmware, in milliseconds.

**partialHeaderTime** (LONGINT): Time taken to write partial headers, in milliseconds.

**clearAbortAreaTime** (LONGINT): Time taken to clear sparecore abort area, in milliseconds.

**writeNodeStatusTime** (LONGINT): Time taken to write the node status information to the sparecore disk, in milliseconds, if any.

## Copyright information

Copyright © 2024 NetApp, Inc. All Rights Reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

LIMITED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (b)(3) of the Rights in Technical Data -Noncommercial Items at DFARS 252.227-7013 (FEB 2014) and FAR 52.227-19 (DEC 2007).

Data contained herein pertains to a commercial product and/or commercial service (as defined in FAR 2.101) and is proprietary to NetApp, Inc. All NetApp technical data and computer software provided under this Agreement is commercial in nature and developed solely at private expense. The U.S. Government has a non-exclusive, non-transferrable, nonsublicensable, worldwide, limited irrevocable license to use the Data only in connection with and in support of the U.S. Government contract under which the Data was delivered. Except as provided herein, the Data may not be used, disclosed, reproduced, modified, performed, or displayed without the prior written approval of NetApp, Inc. United States Government license rights for the Department of Defense are limited to those rights identified in DFARS clause 252.227-7015(b) (FEB 2014).

## Trademark information

NETAPP, the NETAPP logo, and the marks listed at <http://www.netapp.com/TM> are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.