Disks & ZFS

This is information related to the usage of Disks and ZFS

- CLEAR CHECKSUM ERROR IN FREENAS/TRUENAS
- Hard Drive Burn in Testing
- Replacing a Drive Prior to Failure
- Replacing Failed Disks
- Resolving Problems With ZFS
- ZFS commands and status
- ZFS metadata corruption

CLEAR CHECKSUM ERROR IN FREENAS/TRUENAS

https://blog.bianxi.com/2021/10/02/clear-checksum-error-in-freenas-truenas/

Identify error

Errors can be found in TrueNAS Storage section in web page, or use shell in web page, run `zpool status -x` command.

Sample error can be fond in following screen. There are two pools got error. The pool0 got two hard disks, first one got 154 checksum errors, second one got one data error.

```
  pool: pool0
 state: ONLINE
status: One or more devices has experienced an unrecoverable error.  An
    attempt was made to correct the error.  Applications are unaffected.
action: Determine if the device needs to be replaced, and clear the errors
    using 'zpool clear' or replace the device with 'zpool replace'.
   see: https://openzfs.github.io/openzfs-docs/msg/ZFS-8000-9P
  scan: scrub repaired 0B in 00:00:02 with 0 errors on Sat Oct  2 17:39:46 2021
config:

    NAME                                      STATE     READ WRITE CKSUM
    pool0                                     ONLINE       0     0     0
      mirror-0                                ONLINE       0     0     0
        gptid/bf410fcf-2209-11ec-b8aa-001132dbfc9c  ONLINE       0     0   154
        gptid/bfcc498a-2209-11ec-b8aa-001132dbfc9c  ONLINE       0     0     0

errors: No known data errors

  pool: pool01
 state: ONLINE
status: One or more devices has experienced an error resulting in data
```

```
    corruption.  Applications may be affected.
action: Restore the file in question if possible.  Otherwise restore the
    entire pool from backup.
   see: https://openzfs.github.io/openzfs-docs/msg/ZFS-8000-8A
config:

    NAME                                    STATE     READ WRITE CKSUM
    pool01                                  ONLINE       0     0     0
      gptid/75827da1-207a-11ec-afcf-005056a390b2  ONLINE       0     0     1
errors: List of errors unavailable: permission denied

errors: 1 data errors, use '-v' for a list
```

For second error, impacted file can be found using `zpool status -v` command

```
root@truenas[~]# zpool status -v pool01
  pool: pool01
 state: ONLINE
status: One or more devices has experienced an error resulting in data
        corruption.  Applications may be affected.
action: Restore the file in question if possible.  Otherwise restore the
        entire pool from backup.
   see: https://openzfs.github.io/openzfs-docs/msg/ZFS-8000-8A
  scan: scrub repaired 0B in 00:23:22 with 1 errors on Sat Oct  2 21:53:02 2021
config:

        NAME                                    STATE     READ WRITE CKSUM
        pool01                                  ONLINE       0     0     0
          gptid/75827da1-207a-11ec-afcf-005056a390b2  ONLINE       0     0     1

errors: Permanent errors have been detected in the following files:

        /mnt/pool01/download/file.1
root@truenas[~]#
```

Clear error

Run following command to clear the error

```
zpool clear <pool_name>
```

For the pool has data error, which has any file impacted. Delete or overwrite the file.

Then scrub the pool

```
zpool scrub <pool_name>
```

Replace disk

To replace disk, run following command, c0t0d2 is a new disk to replace c0t0d0

```
zpool replace c0t0d0 c0t0d2
```

If the disk replaced at same location, then run following command

```
zpool replace c0t0d0
```

Hard Drive Burn in Testing

https://www.truenas.com/community/resources/hard-drive-burn-in-testing.92/

TESTING FOR HIGH CAPACITY DRIVES WILL TAKE A LONG TIME. EXPECT 12 HOURS OR SO PER TB OF CAPACITY

First of all, the S.M.A.R.T. tests. The first thing that someone unfamiliar with S.M.A.R.T. tests might find strange is the fact that no results are shown when you run the test. The way these tests work is that you initiate the test, it goes off and does its thing, then it records the results for you to check later. So, if this is an initial burn-in test for your entire system, you can initiate tests on all of the drives simultaneously by simply issuing the test command for each drive one after another.

The first test to run is a short self-test:

Code:

```
smartctl -t short /dev/adaX
```

It should indicate that the test will take about 5 minutes. You can immediately begin the same test on the next drive, but you can only run one test on each drive at a time. Once it has completed, run a conveyance test:

Code:

```
smartctl -t conveyance /dev/adaX
```

Again, wait for the test to complete (about 2 minutes this time). Finally, a long test:

Code:

```
smartctl -t long /dev/adaX
```

Note added by @wblock 2018-01-10: this section recommended enabling the kern.geom.debugflags sysctl. Many people still think it has something to do with allowing raw writes. It does not. Instead, it disables a safety system that is intended to prevent writes to disks that are in use (say, by having a mounted filesystem). From man 4 geom:

> 0x10 (allow foot shooting)
> Allow writing to Rank 1 providers. This would, for example,
> allow the super-user to overwrite the MBR on the root disk or
> write random sectors elsewhere to a mounted disk. The
> implications are obvious.

To summarize, this option should generally not be needed. It only makes it possible to harm data. Any disk you are going to overwrite with data should not be mounted or have anything you wish to keep. In fact, best practice is to not be erasing or stress-testing drives on a system that has actual data on it. Since those disks will not have mounted filesystems, this sysctl will not affect being able to write to them. In fact, it will only make it possible to blow away things that are in use.

Now, before we can perform raw disk I/O, we need to enable the kernel geometry debug flags.

This carries some inherent risk, and should probably not be done on a production system. This does not survive through a reboot, so when you're done, just reboot the machine to disable it:
Code:

```
sysctl kern.geom.debugflags=0x10
```

Now that we can execute raw I/O, run a badblocks r/w test.
Unlike the S.M.A.R.T. tests, badblocks runs in the foreground, so once you start it, you won't be able to use the console until the test completes. It also means that if you start it over SSH and lose your connection, the test will be canceled. The answer to this is to use a utility called tmux:
Code:

```
tmux
```

You should now see a green stripe at the bottom of the screen. Now, we can run badblocks. **THIS TEST WILL DESTROY ANY DATA ON THE DISK SO ONLY RUN THIS ON A NEW DISK WITHOUT DATA ON IT OR BACK UP ANY DATA FIRST**:

Code:

```
badblocks -ws /dev/adaX
```

badblocks also offers a non-destructive read-write test that (in theory) shouldn't damage any existing data, but if you do choose to run it on a production drive and suffer data loss, on your own head be it:

Code:

```
badblocks -ns /dev/adaX
```

It has been brought to my attention that badblocks has some limitations with larger drives >2TB. The easy workaround is to manually specify a larger block size for the test.

Code:

```
badblocks -b 4096 -ws /dev/adaX
```

or

Code:

```
badblocks -b 4096 -ns /dev/adaX
```

Once you've started the first test, press Ctrl+B, then " (the double-quote key, not the single quote twice). You should now see a half-white, half-green line through the screen (in PuTTY, it's q's instead of a line, but same thing) with the test continuing in the top half of the screen and a new shell prompt in the bottom. Run the badblocks command again on the next disk, then press Ctrl+B, " again to create another shell. Continue until you've started a test on each disk. If you are connecting over SSH and your session gets disconnected, all of the tests will continue running. When you reconnect, to resume the session and view the test status, simply type:

Code:

```
tmux attach
```

As with the S.M.A.R.T. tests, you can only run one test at a time per drive, but you can test all of your drives simultaneously. In my experience, the tests run just as fast with all drives testing as with a single drive, so for your initial burn-in, there's really no reason not to test all of the drives at once. Also, be prepared for this test to take a very long time, as it is basically the "meat and potatoes" of your burn-in process. For reference, the default 4-pass r/w test took a little over 24 hours on my WD Red 2TB drives, YMMV.

Because S.M.A.R.T. tests only passively detect errors after you've actually attempted to read or write a bad sector, you should run the S.M.A.R.T. long test again after badblocks completes:

Code:

```
smartctl -t long /dev/adaX
```

At this point, you have fully tested all of your drives, and now it's time to view the results of the various S.M.A.R.T. tests:

Code:

```
smartctl -A /dev/adaX
```

This should produce something like this (sorry for the formatting fail):

Code:

```
[root@freenas] ~# smartctl -A /dev/ada0
smartctl 6.2 2013-07-26 r3841 [FreeBSD 9.2-RELEASE-p4 amd64] (local build)
Copyright (C) 2002-13, Bruce Allen, Christian Franke, www.smartmontools.org

=== START OF READ SMART DATA SECTION ===
SMART Attributes Data Structure revision number: 16
Vendor Specific SMART Attributes with Thresholds:
ID# ATTRIBUTE_NAME          FLAG     VALUE WORST THRESH TYPE      UPDATED  WHEN_FAILED RAW_VALUE
  1 Raw_Read_Error_Rate     0x002f   200   200   051    Pre-fail  Always       -       0
  3 Spin_Up_Time            0x0027   175   174   021    Pre-fail  Always       -       4208
  4 Start_Stop_Count        0x0032   100   100   000    Old_age   Always       -       9
  5 Reallocated_Sector_Ct   0x0033   200   200   140    Pre-fail  Always       -       0
  7 Seek_Error_Rate         0x002e   200   200   000    Old_age   Always       -       0
  9 Power_On_Hours          0x0032   100   100   000    Old_age   Always       -       357
10 Spin_Retry_Count         0x0032   100   253   000    Old_age   Always       -       0
11 Calibration_Retry_Count 0x0032   100   253   000    Old_age   Always       -       0
12 Power_Cycle_Count        0x0032   100   100   000    Old_age   Always       -       9
192 Power-Off_Retract_Count 0x0032   200   200   000    Old_age   Always       -       4
193 Load_Cycle_Count        0x0032   200   200   000    Old_age   Always       -       9
194 Temperature_Celsius     0x0022   119   113   000    Old_age   Always       -       28
196 Reallocated_Event_Count 0x0032   200   200   000    Old_age   Always       -       0
197 Current_Pending_Sector  0x0032   200   200   000    Old_age   Always       -       0
198 Offline_Uncorrectable   0x0030   100   253   000    Old_age   Offline      -       0
199 UDMA_CRC_Error_Count    0x0032   200   200   000    Old_age   Always       -       0
200 Multi_Zone_Error_Rate   0x0008   200   200   000    Old_age   Offline      -       0
```

Some of the more important fields right now include the Reallocated_Sector_Ct, Current_Pending_Sector, and Offline_Uncorrectable lines. All of these should have a RAW_VALUE of 0. I'm not sure why the VALUE field is listed as 200, but as long as the RAW_VALUE for each of these fields is 0, that means there are currently no bad sectors. Any result greater than 0 on a new drive should be cause for an immediate RMA.

Once all of your tests have completed, you should reboot your system to disable the kernel geometry debug flags.

Replacing a Drive Prior to Failure

If you want to replace a drive prior to failure, follow this guide. To replace a drive post failure see here Replacing Failed Disks

This guide is for TrueNAS Cobia

- Attach the new drive to the system
- Navigate to Storage > Pool > Manage Devices > Click on the VDEV containing the disk you want to replace > Select the disk > Select the Replace option > Select the new disk
- Wait for the resilver process to complete
- Run a full scrub of the pool
- Shut down the system
- Remove the old disk
- Start the system again

Replacing Failed Disks

This process is for a failed disk. For replacing a disk that has not failed see here Replacing a Drive Prior to Failure

Replacing a Disk

Another disk of the same or greater capacity is required to replace a failed disk. This disk must be installed in the TrueNAS system, not part of an existing storage pool, and available to use as a replacement. The replacement process wipes any data on the replacement disk.

Can I replace a disk in a GELI-encrypted (Legacy) pool?

The TrueNAS **Pool** widget on the main **Dashboard** shows when a disk failure degrades a pool.

Image not found or type unknown

Figure 1: Degraded pool on dashboard widget

Click the *settings* on the pool card to go to the **Storage > Pools > Pool Status** screen to locate the failed disk.

To replace a disk:

1. Take the disk offline.
2. Remove, or replace the disk.
3. Refresh the screen.
4. Bring the disk online.

Taking a Failed Disk Offline

Clicking *more_vert* for the failed disk to show the disk options.

Image not found or type unknown

Figure 2: Pool Status disk options

We recommend you take the disk offline before starting the replacement. This removes the device from the pool and can prevent swap issues. To offline a disk:

Go to **Storage > Pools** screen, click on the *settings* settings icon, and then select **Status** to Open the **Pool Status** screen and display the disks in the pools.

Click the *more_vert* icon for the disk you plan to remove, and then click **Offline**.

Select **Confirm**, then click **OFFLINE**. The disk should now be offline.

Can I use a disk that is failing but still active?

The offline failed?

When the disk status shows as **Offline**, physically remove the disk from the system.

Offline Disk
Image not found or type unknown

Figure 3: Pool Status disk offline

Replacing a Disk

If the replacement disk is not already physically added to the system, add it now.

If replacing a failed disk with an available disk in the system, click **Replace**, select an available disk from the dropdown list, then click **Replace**.

To update the **Pool Status** screen and show the new disk, click **Refresh**.

In the **Pool Status**, open the options for the offline disk and click **Replace**

Replacing Disk
Image not found or type unknown

Figure 4: Replacing disk screen

Select a new member disk and click **Replace Disk**. The new disk must have the same or greater capacity as the disk you are replacing. The replacement fails when the chosen disk has partitions or data present. To destroy any data on the replacement disk and allow the replacement to continue, set the **Force** option.

When the disk wipe completes and TrueNAS starts replacing the failed disk, the **Pool Status** changes to show the in-progress replacement.

Image not found or type unknown

Figure 5: Pool Status replacing disk

TrueNAS resilvers the pool during the replacement process. For pools with large amounts of data, resilvering can take a long time.

Bringing a New Disk Online

When the resilver completes, the pool status screen updates to show the new disk, and the pool status returns to **Online**.

Image not found or type unknown

Figure 6: Pool Status disk replacement complete

During the failed disk replacement process, take these actions after removing and replacing the physical disk to make that replacement disk available:

1. Go to **Disks** and locate the offline disk
2. Click the *more_vert* icon for the offline disk
3. Click **Online**.

Resolving Problems With ZFS

https://docs.oracle.com/cd/E19253-01/819-5461/gbbuw/index.html

The following sections describe how to identify and resolve problems with your ZFS file systems or storage pools:

- Determining If Problems Exist in a ZFS Storage Pool
- Reviewing `zpool status` Output
- System Reporting of ZFS Error Messages

You can use the following features to identify problems with your ZFS configuration:

- Detailed ZFS storage pool information can be displayed by using the `zpool status` command.
- Pool and device failures are reported through ZFS/FMA diagnostic messages.
- Previous ZFS commands that modified pool state information can be displayed by using the `zpool history` command.

Most ZFS troubleshooting involves the `zpool status` command. This command analyzes the various failures in a system and identifies the most severe problem, presenting you with a suggested action and a link to a knowledge article for more information. Note that the command only identifies a single problem with a pool, though multiple problems can exist. For example, data corruption errors generally imply that one of the devices has failed, but replacing the failed device might not resolve all of the data corruption problems.

In addition, a ZFS diagnostic engine diagnoses and reports pool failures and device failures. Checksum, I/O, device, and pool errors associated with these failures are also reported. ZFS failures as reported by `fmd` are displayed on the console as well as the system messages file. In most cases, the `fmd` message directs you to the `zpool status` command for further recovery instructions.

The basic recovery process is as follows:

- If appropriate, use the `zpool history` command to identify the ZFS commands that preceded the error scenario. For example:

```
# zpool history tank
History for 'tank':
2010-07-15.12:06:50 zpool create tank mirror c0t1d0 c0t2d0 c0t3d0
2010-07-15.12:06:58 zfs create tank/erick
2010-07-15.12:07:01 zfs set checksum=off tank/erick
```

In this output, note that checksums are disabled for the `tank/erick` file system. This configuration is not recommended.

- Identify the errors through the `fmd` messages that are displayed on the system console or in the `/var/adm/messages` file.
- Find further repair instructions by using the `zpool status -x` command.
- Repair the failures, which involves the following steps:
  - Replacing the faulted or missing device and bring it online.
  - Restoring the faulted configuration or corrupted data from a backup.
  - Verifying the recovery by using the `zpool status` **`-x`** command.
  - Backing up your restored configuration, if applicable.

This section describes how to interpret `zpool status` output in order to diagnose the type of failures that can occur. Although most of the work is performed automatically by the command, it is important to understand exactly what problems are being identified in order to diagnose the failure. Subsequent sections describe how to repair the various problems that you might encounter.

Determining If Problems Exist in a ZFS Storage Pool

The easiest way to determine if any known problems exist on a system is to use the `zpool status` **`-x`** command. This command describes only pools that are exhibiting problems. If no unhealthy pools exist on the system, then the command displays the following:

```
# zpool status -x
all pools are healthy
```

Without the **`-x`** flag, the command displays the complete status for all pools (or the requested pool, if specified on the command line), even if the pools are otherwise healthy.

For more information about command-line options to the `zpool status` command, see Querying ZFS Storage Pool Status.

Reviewing `zpool status` Output

The complete `zpool status` output looks similar to the following:

```
# zpool status tank
  pool: tank
 state: DEGRADED
status: One or more devices could not be opened.  Sufficient replicas exist for
        the pool to continue functioning in a degraded state.
action: Attach the missing device and online it using 'zpool online'.
   see: http://www.sun.com/msg/ZFS-8000-2Q
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        DEGRADED     0     0     0
          mirror-0  DEGRADED     0     0     0
            c1t0d0  ONLINE       0     0     0
            c1t1d0  UNAVAIL      0     0     0  cannot open

errors: No known data errors
```

This output is described next:

Overall Pool Status Information

This section in the `zpool status` output contains the following fields, some of which are only displayed for pools exhibiting problems:

`pool`
: Identifies the name of the pool.

`state`

Indicates the current health of the pool. This information refers only to the ability of the pool to provide the necessary replication level.

`status`
: Describes what is wrong with the pool. This field is omitted if no errors are found.

`action`
: A recommended action for repairing the errors. This field is omitted if no errors are found.

`see`
: Refers to a knowledge article containing detailed repair information. Online articles are updated more often than this guide can be updated. So, always reference them for the most up-to-date repair procedures. This field is omitted if no errors are found.

`scrub`
: Identifies the current status of a scrub operation, which might include the date and time that the last scrub was completed, a scrub is in progress, or if no scrub was requested.

`errors`
: Identifies known data errors or the absence of known data errors.

Pool Configuration Information

The `config` field in the `zpool status` output describes the configuration of the devices in the pool, as well as their state and any errors generated from the devices. The state can be one of the following: `ONLINE`, `FAULTED`, `DEGRADED`, `UNAVAIL`, or `OFFLINE`. If the state is anything but `ONLINE`, the fault tolerance of the pool has been compromised.

The second section of the configuration output displays error statistics. These errors are divided into three categories:

- `READ` – I/O errors that occurred while issuing a read request
- `WRITE` – I/O errors that occurred while issuing a write request
- `CKSUM` – Checksum errors, meaning that the device returned corrupted data as the result of a read request

These errors can be used to determine if the damage is permanent. A small number of I/O errors might indicate a temporary outage, while a large number might indicate a permanent problem with the device. These errors do not necessarily correspond to data corruption as interpreted by applications. If the device is in a redundant configuration, the devices might show uncorrectable errors, while no errors appear at the mirror or RAID-Z device level. In such cases, ZFS successfully retrieved the good data and attempted to heal the damaged data from existing replicas.

For more information about interpreting these errors, see Determining the Type of Device Failure.

Finally, additional auxiliary information is displayed in the last column of the `zpool status` output. This information expands on the `state` field, aiding in the diagnosis of failures. If a device is `FAULTED`, this field indicates whether the device is inaccessible or whether the data on the device is corrupted. If the device is undergoing resilvering, this field displays the current progress.

For information about monitoring resilvering progress, see Viewing Resilvering Status.

Scrubbing Status

The scrub section of the `zpool status` output describes the current status of any explicit scrubbing operations. This information is distinct from whether any errors are detected on the system, though this information can be used to determine the accuracy of the data corruption error reporting. If the last scrub ended recently, most likely, any known data corruption has been discovered.

Scrub completion messages persist across system reboots.

For more information about the data scrubbing and how to interpret this information, see Checking ZFS File System Integrity.

Data Corruption Errors

The `zpool status` command also shows whether any known errors are associated with the pool. These errors might have been found during data scrubbing or during normal operation. ZFS maintains a persistent log of all data errors associated with a pool. This log is rotated whenever a complete scrub of the system finishes.

Data corruption errors are always fatal. Their presence indicates that at least one application experienced an I/O error due to corrupt data within the pool. Device errors within a redundant pool do not result in data corruption and are not recorded as part of this log. By default, only the number of errors found is displayed. A complete list of errors and their specifics can be found by using the `zpool status` **-v** option. For example:

```
# zpool status -v
  pool: tank
 state: UNAVAIL
status: One or more devices are faulted in response to IO failures.
action: Make sure the affected devices are connected, then run 'zpool clear'.
   see: http://www.sun.com/msg/ZFS-8000-HC
 scrub: scrub completed after 0h0m with 0 errors on Tue Feb  2 13:08:42 2010
config:

       NAME        STATE     READ WRITE CKSUM
       tank        UNAVAIL      0     0     0  insufficient replicas
         c1t0d0    ONLINE       0     0     0
         c1t1d0    UNAVAIL      4     1     0  cannot open

errors: Permanent errors have been detected in the following files:

/tank/data/aaa
/tank/data/bbb
/tank/data/ccc
```

A similar message is also displayed by `fmd` on the system console and the `/var/adm/messages` file. These messages can also be tracked by using the `fmdump` command.

For more information about interpreting data corruption errors, see Identifying the Type of Data Corruption.

System Reporting of ZFS Error Messages

In addition to persistently tracking errors within the pool, ZFS also displays `syslog` messages when events of interest occur. The following scenarios generate events to notify the administrator:

- **Device state transition** – If a device becomes `FAULTED`, ZFS logs a message indicating that the fault tolerance of the pool might be compromised. A similar message is sent if the device is later brought online, restoring the pool to health.
- **Data corruption** – If any data corruption is detected, ZFS logs a message describing when and where the corruption was detected. This message is only logged the first time it is detected. Subsequent accesses do not generate a message.
- **Pool failures and device failures** – If a pool failure or a device failure occurs, the fault manager daemon reports these errors through `syslog` messages as well as the `fmdump`

command.

If ZFS detects a device error and automatically recovers from it, no notification occurs. Such errors do not constitute a failure in the pool redundancy or in data integrity. Moreover, such errors are typically the result of a driver problem accompanied by its own set of error messages.

ZFS commands and status

Command	What it does
zpool status -v <poolname> zpool status -vLP <poolname>	status of the pool, identifies drives as GUID, shows any data errors adding the -LP output the name of the drive in /dev/sdX format
zpool iostat -v <poolname> zpool status -vLP <poolname>	output the I/O statistics of the pool. Excluding a pool name will show all data for all pools adding the -LP flag will do that same as above
zpool list -v <poolname> zpool list -vLP <poolname>	output status of the pool in question. omitting pool name will show all pools. -LP flag does the same as always.
zpool scrub <poolname>	will scrub the pool detecting data errors. will repair errors if sufficient redundancy exists.

Pool or Device Failure and Recovery

ZFS supports a rich set of mechanisms for handling device failure and data corruption. All metadata and data is checksummed, and ZFS automatically repairs bad data from a good copy when corruption is detected.

In order to take advantage of these features, a pool must make use of some form of redundancy, using either mirrored or raidz groups. While ZFS supports running in a non-redundant configuration, where each root vdev is simply a disk or file, this is strongly discouraged. A single case of bit corruption can render some or all of your data unavailable.

A pool's health status is described by one of four states:

`DEGRADED`
: A pool with one or more failed devices, but the data is still available due to a redundant configuration.

`ONLINE`
: A pool that has all devices operating normally.

`SUSPENDED`
: A pool that is waiting for device connectivity to be restored. A suspended pool remains in the wait state until the device issue is resolved.

`UNAVAIL`

A pool with corrupted metadata, or one or more unavailable devices and insufficient replicas to continue functioning.

The health of the top-level vdev, such as mirror or raidz device, is potentially impacted by the state of its associated vdevs, or component devices. A top-level vdev or component device is in one of the following states:

`DEGRADED`

One or more top-level vdevs is in the degraded state because one or more component devices are offline. Sufficient replicas exist to continue functioning.

One or more component devices is in the degraded or faulted state, but sufficient replicas exist to continue functioning. The underlying conditions are as follows:

- The number of checksum errors exceeds acceptable levels and the device is degraded as an indication that something may be wrong. ZFS continues to use the device as necessary.
- The number of I/O errors exceeds acceptable levels. The device could not be marked as faulted because there are insufficient replicas to continue functioning.

`OFFLINE`

The device was explicitly taken offline by the `zpool offline` command.

`ONLINE`

The device is online and functioning.

`REMOVED`

The device was physically removed while the system was running. Device removal detection is hardware-dependent and may not be supported on all platforms.

`UNAVAIL`

The device could not be opened. If a pool is imported when a device was unavailable, then the device will be identified by a unique identifier instead of its path since the path was never correct in the first place.

ZFS metadata corruption

ZFS performs scrub tasks on a regular basis in which in reads each and every file, and compares that file to the checksum. If the checksum is different from the data on the disk, it will attempt to repair that data using the redundency that has been setup with RAID. If it cannot do this, it will log a corruption error which can be seen using the `zpool status -v <poolname>` command. This error will remain until the file is removed and another scrub task is preformed.

If the file is gone, but snapshots still reference that file, then an identifier will be used such as `<0x00xxx>:<0xXX00>` The zpool will also report as being unhealthy.

If the actual ZFS pool metadata is corrupted, that is another story. If crucial metadata is corrupted then the pool will report as being corrupted, and may even fail to import upon boot. A pool with corrupted metadata should be backed up immediately, destroyed and restored from backup.

ZFS metadata corruption can repot as being `<metadata:0xXXxx>`

Further reading zfs pool metadata corrupt