NAME

ztest - was written by the ZFS Developers as a ZFS unit test

SYNOPSIS

ztest [**-VEG**] [**-v** *vdevs*] [**-s** *size_of_each_vdev*] [**-a** *alignment_shift*] [**-m** *mirror_copies*]
[**-r** *raidz_disks/draid_disks*] [**-R** *raid_parity*] [**-K** *raid_kind*] [**-D** *draid_data*] [**-S** *draid_spares*]
[**-C** *vdev_class_state*] [**-d** *datasets*] [**-t** *threads*] [**-g** *gang_block_threshold*]
[**-i** *initialize_pool_i_times*] [**-k** *kill_percentage*] [**-p** *pool_name*] [**-T** *time*] [**-z** *zil_failure_rate*]

DESCRIPTION

ztest was written by the ZFS Developers as a ZFS unit test. The tool was developed in tandem with the ZFS functionality and was executed nightly as one of the many regression test against the daily build. As features were added to ZFS, unit tests were also added to **ztest**. In addition, a separate test development team wrote and executed more functional and stress tests.

By default **ztest** runs for ten minutes and uses block files (stored in */tmp*) to create pools rather than using physical disks. Block files afford **ztest** its flexibility to play around with zpool components without requiring large hardware configurations. However, storing the block files in */tmp* may not work for you if you have a small tmp directory.

By default is non-verbose. This is why entering the command above will result in **ztest** quietly executing for 5 minutes. The **-V** option can be used to increase the verbosity of the tool. Adding multiple **-V** options is allowed and the more you add the more chatty **ztest** becomes.

After the **ztest** run completes, you should notice many *ztest.** files lying around. Once the run completes you can safely remove these files. Note that you shouldn't remove these files during a run. You can re-use these files in your next **ztest** run by using the **-E** option.

OPTIONS

-h, **-?**, **--help**
    Print a help summary.

-v, **--vdevs**= (default: **5**)
    Number of vdevs.

-s, **--vdev-size**= (default: **64M**)
    Size of each vdev.

-a, **--alignment-shift**= (default: **9**) (use **0** for random)
    Alignment shift used in test.

-m, **--mirror-copies**= (default: **2**)
Number of mirror copies.

-r, **--raid-disks**= (default: **4** for raidz/**16** for draid)
Number of raidz/draid disks.

-R, **--raid-parity**= (default: **1**)
Raid parity (raidz & draid).

-K, **--raid-kind**=**raidz**|**draid**|**random** (default: **random**)
The kind of RAID config to use. With **random** the kind alternates between raidz and draid.

-D, **--draid-data**= (default: **4**)
Number of data disks in a dRAID redundancy group.

-S, **--draid-spares**= (default: **1**)
Number of dRAID distributed spare disks.

-d, **--datasets**= (default: **7**)
Number of datasets.

-t, **--threads**= (default: **23**)
Number of threads.

-g, **--gang-block-threshold**= (default: **32K**)
Gang block threshold.

-i, **--init-count**= (default: **1**)
Number of pool initializations.

-k, **--kill-percentage**= (default: **70%**)
Kill percentage.

-p, **--pool-name**= (default: **ztest**)
Pool name.

-f, **--vdev-file-directory**= (default: */tmp*)
File directory for vdev files.

-M, **--multi-host**

Multi-host; simulate pool imported on remote host.

-E, **--use-existing-pool**
Use existing pool (use existing pool instead of creating new one).

-T, **--run-time**= (default: **300**s)
Total test run time.

-P, **--pass-time**= (default: **60**s)
Time per pass.

-F, **--freeze-loops**= (default: **50**)
Max loops in **spa_freeze**().

-B, **--alt-ztest**=
Path to alternate ("older") **ztest** to drive, which will be used to initialise the pool, and, a stochastic half the time, to run the tests. The parallel *lib* directory is prepended to LD_LIBRARY_PATH; i.e. given **-B** *./chroots/lenny/usr/bin/***ztest**, *./chroots/lenny/usr/lib* will be loaded.

-C, **--vdev-class-state**=**on**|**off**|**random** (default: **random**)
The vdev allocation class state.

-o, **--option**=*variable*=*value*
Set global *variable* to an unsigned 32-bit integer *value* (little-endian only).

-G, **--dump-debug**
Dump zfs_dbgmsg buffer before exiting due to an error.

-V, **--verbose**
Verbose (use multiple times for ever more verbosity).

EXAMPLES

To override */tmp* as your location for block files, you can use the **-f** option:

```
# ztest -f /
```

To get an idea of what **ztest** is actually testing try this:

```
# ztest -f / -VVV
```

Maybe you'd like to run **ztest** for longer? To do so simply use the **-T** option and specify the runlength in seconds like so:

```
# ztest -f / -V -T 120
```

ENVIRONMENT VARIABLES

ZFS_HOSTID=*id*

Use *id* instead of the SPL hostid to identify this host. Intended for use with **ztest**, but this environment variable will affect any utility which uses libzpool, including zpool(8). Since the kernel is unaware of this setting, results with utilities other than ztest are undefined.

ZFS_STACK_SIZE=*stacksize*

Limit the default stack size to *stacksize* bytes for the purpose of detecting and debugging kernel stack overflows. This value defaults to *32K* which is double the default *16K* Linux kernel stack size.

In practice, setting the stack size slightly higher is needed because differences in stack usage between kernel and user space can lead to spurious stack overflows (especially when debugging is enabled). The specified value will be rounded up to a floor of PTHREAD_STACK_MIN which is the minimum stack required for a NULL procedure in user space.

By default the stack size is limited to *256K*.

SEE ALSO

zdb(1), zfs(1), zpool(1), spl(4)