NAME

zpool-create - create ZFS storage pool

SYNOPSIS

zpool create [**-dfn**] [**-m** *mountpoint*] [**-o** *property=value*]<?> [**-o feature@***feature=value*]
[**-o** *compatibility*=**off**|**legacy**|*file*[,*file*]<?>] [**-O** *file-system-property=value*]<?> [**-R** *root*]
[**-t** *tname*] *pool vdev*<?>

DESCRIPTION

Creates a new storage pool containing the virtual devices specified on the command line. The pool name must begin with a letter, and can only contain alphanumeric characters as well as the underscore ("_"), dash ("**-**"), colon ("**:**"), space (" "), and period ("**.**"). The pool names **mirror**, **raidz**, **draid**, **spare** and **log** are reserved, as are names beginning with **mirror**, **raidz**, **draid**, and **spare**. The *vdev* specification is described in the *Virtual Devices* section of zpoolconcepts(7).

The command attempts to verify that each device specified is accessible and not currently in use by another subsystem. However this check is not robust enough to detect simultaneous attempts to use a new device in different pools, even if **multihost**= **enabled**. The administrator must ensure that simultaneous invocations of any combination of **zpool replace**, **zpool create**, **zpool add**, or **zpool labelclear** do not refer to the same device. Using the same device in two pools will result in pool corruption.

There are some uses, such as being currently mounted, or specified as the dedicated dump device, that prevents a device from ever being used by ZFS. Other uses, such as having a preexisting UFS file system, can be overridden with **-f**.

The command also checks that the replication strategy for the pool is consistent. An attempt to combine redundant and non-redundant storage in a single pool, or to mix disks and files, results in an error unless **-f** is specified. The use of differently-sized devices within a single raidz or mirror group is also flagged as an error unless **-f** is specified.

Unless the **-R** option is specified, the default mount point is */pool*. The mount point must not exist or must be empty, or else the root dataset will not be able to be be mounted. This can be overridden with the **-m** option.

By default all supported features are enabled on the new pool. The **-d** option and the **-o** *compatibility* property (e.g **-o compatibility**=*2020*) can be used to restrict the features that are enabled, so that the pool can be imported on other releases of ZFS.

-d Do not enable any features on the new pool. Individual features can be enabled by setting their

corresponding properties to **enabled** with **-o**. See zpool-features(7) for details about feature properties.

-f Forces use of *vdev*s, even if they appear in use or specify a conflicting replication level. Not all devices can be overridden in this manner.

-m *mountpoint*
Sets the mount point for the root dataset. The default mount point is */pool* or *altroot/pool* if **altroot** is specified. The mount point must be an absolute path, **legacy**, or **none**. For more information on dataset mount points, see zfsprops(7).

-n Displays the configuration that would be used without actually creating the pool. The actual pool creation can still fail due to insufficient privileges or device sharing.

-o *property=value*
Sets the given pool properties. See zpoolprops(7) for a list of valid properties that can be set.

-o *compatibility*=**off**|**legacy**|*file*[,*file*]<?>
Specifies compatibility feature sets. See zpool-features(7) for more information about compatibility feature sets.

-o feature@*feature=value*
Sets the given pool feature. See the zpool-features(7) section for a list of valid features that can be set. Value can be either disabled or enabled.

-O *file-system-property=value*
Sets the given file system properties in the root file system of the pool. See zfsprops(7) for a list of valid properties that can be set.

-R *root* Equivalent to **-o cachefile**=**none -o altroot**=*root*

-t *tname* Sets the in-core pool name to *tname* while the on-disk name will be the name specified as *pool*. This will set the default of the **cachefile** property to **none**. This is intended to handle name space collisions when creating pools for other systems, such as virtual machines or physical machines whose pools live on network block devices.

EXAMPLES

Example 1: Creating a RAID-Z Storage Pool

The following command creates a pool with a single raidz root vdev that consists of six disks:

zpool create *tank* **raidz** *sda sdb sdc sdd sde sdf*

Example 2: Creating a Mirrored Storage Pool

The following command creates a pool with two mirrors, where each mirror contains two disks:

zpool create *tank* **mirror** *sda sdb* **mirror** *sdc sdd*

Example 3: Creating a ZFS Storage Pool by Using Partitions

The following command creates a non-redundant pool using two disk partitions:

zpool create *tank sda1 sdb2*

Example 4: Creating a ZFS Storage Pool by Using Files

The following command creates a non-redundant pool using files. While not recommended, a pool based on files can be useful for experimental purposes.

zpool create *tank /path/to/file/a /path/to/file/b*

Example 5: Managing Hot Spares

The following command creates a new pool with an available hot spare:

zpool create *tank* **mirror** *sda sdb* **spare** *sdc*

Example 6: Creating a ZFS Pool with Mirrored Separate Intent Logs

The following command creates a ZFS storage pool consisting of two, two-way mirrors and mirrored log devices:

zpool create *pool* **mirror** *sda sdb* **mirror** *sdc sdd* **log mirror** *sde sdf*

SEE ALSO

zpool-destroy(8), zpool-export(8), zpool-import(8)