NAME

hastctl - Highly Available Storage control utility

SYNOPSIS

hastctl create [**-d**] [**-c** *config*] [**-e** *extentsize*] [**-k** *keepdirty*] [**-m** *mediasize*] *name ...*
hastctl role [**-d**] [**-c** *config*] <init | primary | secondary> *all* | *name ...*
hastctl list [**-d**] [**-c** *config*] [*all* | *name ...*]
hastctl status [**-d**] [**-c** *config*] [*all* | *name ...*]
hastctl dump [**-d**] [**-c** *config*] [*all* | *name ...*]

DESCRIPTION

The **hastctl** utility is used to control the behaviour of the hastd(8) daemon.

This utility should be used by HA software like **heartbeat** or **ucarp** to setup HAST resources role when changing from primary mode to secondary or vice versa. Be aware that if a file system like UFS exists on HAST provider and primary node dies, file system has to be checked for inconsistencies with the fsck(8) utility after switching secondary node to primary role.

The first argument to **hastctl** indicates an action to be performed:

create Initialize local provider configured for the given resource. Additional options include:

-e *extentsize* Size of an extent. Extent is a block which is used for synchronization. hastd(8) maintains a map of dirty extents and extent is the smallest region that can be marked as dirty. If any part of an extent is modified, entire extent will be synchronized when nodes connect. If extent size is too small, there will be too much disk activity related to dirty map updates, which will degrade performance of the given resource. If extent size is too large, synchronization, even in case of short outage, can take a long time increasing the risk of losing up-to-date node before synchronization process is completed. The default extent size is *2MB*.

-k *keepdirty* Maximum number of dirty extents to keep dirty all the time. Most recently used extents are kept dirty to reduce number of metadata updates. The default number of most recently used extents which will be kept dirty is *64*.

-m *mediasize* Size of the smaller provider used as backend storage on both nodes. This option can be omitted if node providers have the same size on both sides.

If size is suffixed with a k, M, G or T, it is taken as a kilobyte, megabyte, gigabyte or terabyte measurement respectively.

role Change role of the given resource. The role can be one of:

init Resource is turned off.

primary Local hastd(8) daemon will act as primary node for the given resource. System on which resource role is set to primary can use */dev/hast/<name>* GEOM provider.

secondary Local hastd(8) daemon will act as secondary node for the given resource - it will wait for connection from the primary node and will handle I/O requests received from it. GEOM provider */dev/hast/<name>* will not be created on secondary node.

list Present verbose status of the configured resources.

status Present terse (and more easy machine-parseable) status of the configured resources.

dump Dump metadata stored on local component for the configured resources.

In addition, every subcommand can be followed by the following options:

-c *config* Specify alternative location of the configuration file. The default location is */etc/hast.conf*.

-d Print debugging information. This option can be specified multiple times to raise the verbosity level.

FILES

/etc/hast.conf Configuration file for **hastctl** and hastd(8).
/var/run/hastctl Control socket used by **hastctl** to communicate with the hastd(8) daemon.

EXIT STATUS

Exit status is 0 on success, or one of the values described in sysexits(3) on failure.

EXAMPLES

Initialize HAST provider, create file system on it and mount it.

```
nodeB# hastctl create shared
nodeB# hastd
nodeB# hastctl role secondary shared

nodeA# hastctl create shared
nodeA# hastd
```

```
nodeA# hastctl role primary shared
nodeA# newfs -U /dev/hast/shared
nodeA# mount -o noatime /dev/hast/shared /shared
nodeA# application_start
```

Switch roles for the **shared** HAST resource.

```
nodeA# application_stop
nodeA# umount -f /shared
nodeA# hastctl role secondary shared

nodeB# hastctl role primary shared
nodeB# fsck -t ufs /dev/hast/shared
nodeB# mount -o noatime /dev/hast/shared /shared
nodeB# application_start
```

SEE ALSO

sysexits(3), geom(4), hast.conf(5), fsck(8), ggatec(8), ggatel(8), hastd(8), mount(8), newfs(8)

HISTORY

The **hastctl** utility appeared in FreeBSD 8.1.

AUTHORS

The **hastctl** was developed by Pawel Jakub Dawidek <*pjd@FreeBSD.org*> under sponsorship of the FreeBSD Foundation.