btrfs-qgroup(8)

SYNOPSIS

btrfs qgroup <subcommand> <args>

DESCRIPTION

btrfs qgroup is used to control quota group (qgroup) of a btrfs filesystem.

Note

To use qgroup you need to enable quota first using **btrfs quota enable** command.

Warning

Qgroup is not stable yet and will impact performance in current mainline kernel (v4.14).

QGROUP

Quota groups or qgroup in btrfs make a tree hierarchy, the leaf qgroups are attached to subvolumes. The size limits are set per qgroup and apply when any limit is reached in tree that contains a given subvolume.

The limits are separated between shared and exclusive and reflect the extent ownership. For example a fresh snapshot shares almost all the blocks with the original subvolume, new writes to either subvolume will raise towards the exclusive limit.

Note

Qgroup limit only works when qgroup is in a consistent state. If some workload marks qgroup inconsistent (like assigning a qgroup to another qgroup), the limit will no longer work until the inconsistent flag is cleared by **btrfs quota rescan**.

The qgroup identifiers conform to *level/id* where level 0 is reserved to the qgroups associated with subvolumes. Such qgroups are created automatically.

The qgroup hierarchy is built by commands **create** and **assign**.

Note

If the qgroup of a subvolume is destroyed, quota about the subvolume will not be functional until qgroup *0/<subvolume id>* is created again.

SUBCOMMAND

assign [options] <src> <dst> <path>

Assign qgroup *src* as the child qgroup of *dst* in the btrfs filesystem identified by *path*.

`Options`

`--rescan`

(default since: 4.19) Automatically schedule quota rescan if the new qgroup assignment would lead to quota inconsistency. See *QUOTA RESCAN* for more information.

`--no-rescan`

Explicitly ask not to do a rescan, even if the assignment will make the quotas inconsistent. This may be useful for repeated calls where the rescan would add unnecessary overhead.

create <qgroupid> <path>

Create a subvolume quota group.

For the *0/<subvolume id>* qgroup, a qgroup can be created even before the subvolume is created.

destroy <qgroupid> <path>

Destroy a qgroup.

If a qgroup is not isolated, meaning it is a parent or child qgroup, then it can only be destroyed after the relationship is removed.

clear-stale <path>

Clear all stale qgroups whose subvolume does not exist anymore, this is the level 0 qgroup like 0/subvolid. Higher level qgroups are not deleted even if they don't have any child qgroups.

limit [options] <size>|none [<qgroupid>] <path>

Limit the size of a qgroup to *size* or no limit in the btrfs filesystem identified by *path*.

If *qgroupid* is not given, qgroup of the subvolume identified by *path* is used if possible.

`Options`

`-c`

limit amount of data after compression. This is the default, it is currently not possible to turn off this option.

`-e`

limit space exclusively assigned to this qgroup.

remove <src> <dst> <path>

Remove the relationship between child qgroup *src* and parent qgroup *dst* in the btrfs filesystem identified by *path*.

`Options`

`--rescan`

(default since: 4.19) Automatically schedule quota rescan if the removed qgroup relation would lead to quota inconsistency. See *QUOTA RESCAN* for more information.

`--no-rescan`

Explicitly ask not to do a rescan, even if the removal will make the quotas inconsistent. This may be useful for repeated calls where the rescan would add unnecessary overhead.

show [options] <path>

Show all qgroups in the btrfs filesystem identified by <path>.

`Options`

`-p`

print parent qgroup id.

`-c`

print child qgroup id.

`-r`

print limit of referenced size of qgroup.

`-e`

print limit of exclusive size of qgroup.

`-F`

list all qgroups which impact the given path(include ancestral qgroups)

`-f`

list all qgroups which impact the given path(exclude ancestral qgroups)

`--raw`

raw numbers in bytes, without the *B* suffix.

`--human-readable`

print human friendly numbers, base 1024, this is the default

`--iec`

select the 1024 base for the following options, according to the IEC standard.

`--si`

select the 1000 base for the following options, according to the SI standard.

`--kbytes`

show sizes in KiB, or kB with --si.

`--mbytes`

show sizes in MiB, or MB with --si.

`--gbytes`

show sizes in GiB, or GB with --si.

`--tbytes`

show sizes in TiB, or TB with --si.

--sort=[+/-]<attr>[,[+/-]<attr>]...

list qgroups in order of <attr>.

<attr> can be one or more of qgroupid,rfer,excl,max_rfer,max_excl.

Prefix + means ascending order and - means descending order of *attr*. If no prefix is given, use ascending order by default.

If multiple *attr* values are given, use comma to separate.

`--sync`

To retrieve information after updating the state of qgroups, force sync of the filesystem identified by *path* before getting information.

SPECIAL PATHS

For *btrfs qgroup show* subcommand, the `path` column may has some special strings:

<toplevel>

The toplevel subvolume

<under deletion>

The subvolume has been deleted (it's directory removed), but the subvolume metadata not not yet fully cleaned.

<squota space holder>

For simple quota mode only. By its design, a fully deleted subvolume may still have accounting on it, so even the subvolume is gone, the numbers are still here for future accounting.

<stale>

The qgroup has no corresponding subvolume anymore, and the qgroup can be cleaned up under most cases. The only exception is that, if the qgroup numbers are inconsistent and the qgroup numbers are not all zeros, some older kernels may refuse to delete such qgroups until a full rescan.

QUOTA RESCAN

The rescan reads all extent sharing metadata and updates the respective qgroups accordingly.

The information consists of bytes owned exclusively (*excl*) or shared/referred to (*rfer*). There's no explicit information about which extents are shared or owned exclusively. This means when qgroup relationship changes, extent owners change and qgroup numbers are no longer consistent unless we do a full rescan.

However there are cases where we can avoid a full rescan, if a subvolume whose *rfer* number equals its *excl* number, which means all bytes are exclusively owned, then assigning/removing this subvolume only needs to add/subtract *rfer* number from its parent qgroup. This can speed up the rescan.

EXAMPLES

Make a parent group that has two quota group children

Given the following filesystem mounted at `/mnt/my-vault`

```
Label: none  uuid: 60d2ab3b-941a-4f22-8d1a-315f329797b2
       Total devices 1 FS bytes used 128.00KiB
       devid    1 size 5.00GiB used 536.00MiB path /dev/vdb
```

Enable quota and create subvolumes. Check subvolume ids.

```
$ cd /mnt/my-vault
$ btrfs quota enable .
$ btrfs subvolume create a
$ btrfs subvolume create b
$ btrfs subvolume list .

ID 261 gen 61 top level 5 path a
ID 262 gen 62 top level 5 path b
```

Create qgroup and set limit to 10MiB.

```
$ btrfs qgroup create 1/100 .
$ btrfs qgroup limit 10M 1/100 .
$ btrfs qgroup assign 0/261 1/100 .
$ btrfs qgroup assign 0/262 1/100 .
```

And check qgroups.

```
$ btrfs qgroup show .

qgroupid         rfer         excl
--------         ----         ----
0/5          16.00KiB     16.00KiB
0/261        16.00KiB     16.00KiB
0/262        16.00KiB     16.00KiB
1/100        32.00KiB     32.00KiB
```

EXIT STATUS

btrfs qgroup returns a zero exit status if it succeeds. Non zero is returned in case of failure.

AVAILABILITY

btrfs is part of btrfs-progs. Please refer to the documentation at https://btrfs.readthedocs.io.

stable

SEE ALSO

btrfs-quota(8), btrfs-subvolume(8), mkfs.btrfs(8)