NAME

dmstats — device-mapper statistics management

SYNOPSIS

dmsetup stats *command* [OPTIONS]

dmstats *command device_name* | **--major** *major* **--minor** *minor* | **-u|--uuid** *uuid* [**-v|--verbose]**
dmstats clear *device_name* [**--allprograms|--programid** *id*] [**--allregions|--regionid** *id*]
dmstats create *device_name...*|*file_path...*|**--alldevices** [**--areas** *nr_areas*|**--areasize** *area_size*]
[**--bounds** *histogram_boundaries*] [**--filemap**] [**--follow** *follow_mode*] [**--foreground**]
[**--nomonitor**] [**--nogroup**] [**--precise**] [**--start** *start_sector* **--length** *length*|**--segments**]
[**--userdata** *user_data*] [**--programid** *id*]
dmstats delete *device_name*|**--alldevices** [**--allprograms|--programid** *id*] [**--allregions|--regionid**
id]
dmstats group [*device_name*|**--alldevices**] [**--alias** *name*] [**--regions** *regions*]
dmstats help [**-c|-C|--columns**]
dmstats list [*device_name*] [**--histogram**] [**--allprograms|--programid** *id*] [**--units** *units*] [**--area**]
[**--region**] [**--group**] [**--nosuffix**] [**--notimesuffix**] [**-v|--verbose]**
dmstats print [*device_name*] [**--clear**] [**--allprograms|--programid** *id*] [**--allregions|--regionid** *id*]
dmstats report [*device_name*] [**--interval** *seconds*] [**--count** *count*] [**--units** *units*] [**--histogram**]
[**--allprograms|--programid** *id*] [**--allregions|--regionid** *id*] [**--area**] [**--region**] [**--group**]
[**-O|--sort** *sort_fields*] [**-S|--select** *selection*] [**--units** *units*] [**--nosuffix**] [**--notimesuffix**]
dmstats ungroup [*device_name*|**--alldevices**] [**--groupid** *id*]
dmstats update_filemap *file_path* [**--groupid** *id*] [**--follow** *follow_mode*] [**--foreground**]

DESCRIPTION

The dmstats program manages IO statistics regions for devices that use the device-mapper driver. Statistics regions may be created, deleted, listed and reported on using the tool.

The first argument to dmstats is a *command*.

The second argument is the *device name*, *uuid* or *major* and *minor* numbers.

Further options permit the selection of regions, output format control, and reporting behaviour.

When no device argument is given dmstats will by default operate on all device-mapper devices present. The **create** and **delete** commands require the use of **--alldevices** when used in this way.

OPTIONS

--alias *name*
Specify an alias name for a group.

--alldevices
If no device arguments are given allow operation on all devices when creating or deleting regions.

--allprograms
Include regions from all program IDs for list and report operations.

--allregions
Include all present regions for commands that normally accept a single region identifier.

--area
When peforming a list or report, include objects of type area in the results.

--areas *nr_areas*
Specify the number of statistics areas to create within a new region.

--areasize *area_size*[**b|B|s|S|k|K|m|M|g|G|t|T|p|P|e|E**]
Specify the size of areas into which a new region should be divided. An optional suffix selects units of: (**b**)ytes, (**s**)ectors, (**k**)ilobytes, (**m**)egabytes, (**g**)igabytes, (**t**)erabytes, (**p**)etabytes,

(**e**)xabytes. Capitalise to use multiples of 1000 (S.I.) instead of 1024.

−−clear
When printing statistics counters, also atomically reset them to zero.

−−count *count*
Specify the iteration count for repeating reports. If the count argument is zero reports will continue to repeat until interrupted.

−−group
When peforming a list or report, include objects of type group in the results.

−−filemap
Instead of creating regions on a device as specified by command line options, open the file found at each **file_path** argument, and create regions corresponding to the locations of the on-disk extents allocated to the file(s).

−−nomonitor
Disable the **dmfilemapd** daemon when creating new file mapped groups. Normally the device-mapper filemap monitoring daemon, **dmfilemapd**, is started for each file mapped group to update the set of regions as the file changes on-disk: use of this option disables this behaviour.

Regions in the group may still be updated with the **update_filemap** command, or by starting the daemon manually.

−−follow *follow_mode*
Specify the **dmfilemapd** file following mode. The file map monitoring daemon can monitor files in two distinct ways: the mode affects the behaviour of the daemon when a file under monitoring is renamed or unlinked, and the conditions which cause the daemon to terminate.

The **follow_mode** argument is either "inode", for follow-inode mode, or "path", for follow-path.

If follow-inode mode is used, the daemon will hold the file open, and continue to update regions from the same file descriptor. This means that the mapping will follow rename, move (within the same file system), and unlink operations. This mode is useful if the file is expected to be moved, renamed, or unlinked while it is being monitored.

In follow-inode mode, the daemon will exit once it detects that the file has been unlinked and it is the last holder of a reference to it.

If follow-path is used, the daemon will re-open the provided path on each monitoring iteration. This means that the group will be updated to reflect a new file being moved to the same path as the original file. This mode is useful for files that are expected to be updated via unlink and rename.

In follow-path mode, the daemon will exit if the file is removed and not replaced within a brief tolerance interval.

In either mode, the daemon exits automatically if the monitored group is removed.

−−foreground
Specify that the **dmfilemapd** daemon should run in the foreground. The daemon will not fork into the background, and will replace the **dmstats** command that started it.

−−groupid *id*
Specify the group to operate on.

−−bounds *histogram_boundaries*[**ns**|**us**|**ms**|**s**]
Specify the boundaries of a latency histogram to be tracked for the region as a comma separated list of latency values. Latency values are given in nanoseconds. An optional unit suffix of **ns**, **us**, **ms**, or **s** may be given after each value to specify units of nanoseconds, microseconds, miliseconds

or seconds respectively.

−−histogram
When used with the **report** and **list** commands select default fields that emphasize latency histogram data.

−−interval *seconds*
Specify the interval in seconds between successive iterations for repeating reports. If **−−interval** is specified but **−−count** is not, reports will continue to repeat until interrupted.

−−length *length*[**b**|**B**|**s**|**S**|**k**|**K**|**m**|**M**|**g**|**G**|**t**|**T**|**p**|**P**|**e**|**E**]
Specify the length of a new statistics region in sectors. An optional suffix selects units of: (**b**)ytes, (**s**)ectors, (**k**)ilobytes, (**m**)egabytes, (**g**)igabytes, (**t**)erabytes, (**p**)etabytes, (**e**)xabytes. Capitalise to use multiples of 1000 (S.I.) instead of 1024.

−j|−−major *major*
Specify the major number.

−m|−−minor *minor*
Specify the minor number.

−−nogroup
When creating regions mapping the extents of a file in the file system, do not create a group or set an alias.

−−nosuffix
Suppress the suffix on output sizes. Use with **−−units** (except h and H) if processing the output.

−−notimesuffix
Suppress the suffix on output time values. Histogram boundary values will be reported in units of nanoseconds.

−o|−−options
Specify which report fields to display.

−O|−−sort *sort_fields*
Sort output according to the list of fields given. Precede any sort field with '**−**' for a reverse sort on that column.

−−precise
Attempt to use nanosecond precision counters when creating new statistics regions.

−−programid *id*
Specify a program ID string. When creating new statistics regions this string is stored with the region. Subsequent operations may supply a program ID in order to select only regions with a matching value. The default program ID for dmstats-managed regions is "dmstats".

−−region
When peforming a list or report, include objects of type region in the results.

−−regionid *id*
Specify the region to operate on.

−−regions *region_list*
Specify a list of regions to group. The group list is a comma-separated list of region identifiers. Continuous sequences of identifiers may be expressed as a hyphen separated range, for example: '1−10'.

−−relative
If displaying the histogram report show relative (percentage) values instead of absolute counts.

−S|−−select *selection*
Display only rows that match *selection* criteria. All rows with the additional "selected" column (**−o selected**) showing 1 if the row matches the *selection* and 0 otherwise. The selection criteria are defined by specifying column names and their valid values while making use of supported

comparison operators.

−−start *start*[**b**|**B**|**s**|**S**|**k**|**K**|**m**|**M**|**g**|**G**|**t**|**T**|**p**|**P**|**e**|**E**]
Specify the start offset of a new statistics region in sectors. An optional suffix selects units of: (**b**)ytes, (**s**)ectors, (**k**)ilobytes, (**m**)egabytes, (**g**)igabytes, (**t**)erabytes, (**p**)etabytes, (**e**)xabytes. Capitalise to use multiples of 1000 (S.I.) instead of 1024.

−−segments
When used with **create**, create a new statistics region for each target contained in the given device(s). This causes a separate region to be allocated for each segment of the device.

The newly created regions are automatically placed into a group unless the **−−nogroup** option is given. When grouping is enabled a group alias may be specified using the **−−alias** option.

−−units [*units*][**h**|**H**|**b**|**B**|**s**|**S**|**k**|**K**|**m**|**M**|**g**|**G**|**t**|**T**|**p**|**P**|**e**|**E**]
Set the display units for report output. All sizes are output in these units: (**h**)uman-readable, (**b**)ytes, (**s**)ectors, (**k**)ilobytes, (**m**)egabytes, (**g**)igabytes, (**t**)erabytes, (**p**)etabytes, (**e**)xabytes. Capitalise to use multiples of 1000 (S.I.) instead of 1024. Can also specify custom units e.g. **−−units 3M**.

−−userdata *user_data*
Specify user data (a word) to be stored with a new region. The value is added to any internal auxilliary data (for example, group information), and stored with the region in the aux_data field provided by the kernel. Whitespace is not permitted.

−u|**−−uuid**
Specify the uuid.

−v|**−−verbose** [**−v**|**−−verbose**]
Produce additional output.

COMMANDS

clear *device_name* [**−−allprograms**|**−−programid** *id*] [**−−allregions**|**−−regionid** *id*]
Instructs the kernel to clear statistics counters for the speficied regions (with the exception of in-flight IO counters).

create *device_name...*|*file_path...*|**−−alldevices** [**−−areas** *nr_areas*|**−−areasize** *area_size*] [**−−bounds** *histogram_boundaries*] [**−−filemap**] [**−−follow** *follow_mode*] [**−−foreground**] [**−−nomonitor**] [**−−nogroup**] [**−−precise**] [**−−start** *start_sector* **−−length** *length*|**−−segments**] [**−−userdata** *user_data*] [**−−programid** *id*]
Creates one or more new statistics regions on the specified device(s).

The region will span the entire device unless **−−start** and **−−length** or **−−segments** are given. The **−−start** an **−−length** options allow a region of arbitrary length to be placed at an arbitrary offset into the device. The **−−segments** option causes a new region to be created for each target in the corresponding device-mapper device's table.

If the **−−precise** option is used the command will attempt to create a region using nanosecond precision counters.

If **−−bounds** is given a latency histogram will be tracked for the new region. The boundaries of the histogram bins are given as a comma separated list of latency values. There is an implicit lower bound of zero on the first bin and an implicit upper bound of infinity (or the configured interval duration) on the final bin.

Latencies are given in nanoseconds. An optional unit suffix of ns, us, ms, or s may be given after each value to specify units of nanoseconds, microseconds, miliseconds or seconds respectively, so for example, 10ms is equivalent to 10000000. Latency values with a precision of less than one milisecond can only be used when precise timestamps are enabled: if **−−precise** is not given and

values less than one milisecond are used it will be enabled automatically.

An optional **program_id** or **user_data** string may be associated with the region. A **program_id** may then be used to select regions for subsequent list, print, and report operations. The **user_data** stores an arbitrary string and is not used by dmstats or the device-mapper kernel statistics subsystem.

By default dmstats creates regions with a **program_id** of "dmstats".

On success the **region_id** of the newly created region is printed to stdout.

If the **−−filemap** option is given with a regular file, or list of files, as the **file_path** argument, instead of creating regions with parameters specified on the command line, **dmstats** will open the files located at **file_path** and create regions corresponding to the physical extents allocated to the file. This can be used to monitor statistics for individual files in the file system, for example, virtual machine images, swap areas, or large database files.

To work with the **−−filemap** option, files must be located on a local file system, backed by a device-mapper device, that supports physical extent data using the FIEMAP ioctl (Ext4 and XFS for e.g.).

By default regions that map a file are placed into a group and the group alias is set to the basename of the file. This behaviour can be overridden with the **−−alias** and **−−nogroup** options.

Creating a group that maps a file automatically starts a daemon, **dmfilemapd** to monitor the file and update the mapping as the extents allocated to the file change. This behaviour can be disabled using the **−−nomonitor** option.

Use the **−−group** option to only display information for groups when listing and reporting.

delete *device_name*|**−−alldevices** [**−−allprograms**|**−−programid** *id*] [**−−allregions**|**−−regionid** *id*]
Delete the specified statistics region. All counters and resources used by the region are released and the region will not appear in the output of subsequent list, print, or report operations.

All regions registered on a device may be removed using **−−allregions**.

To remove all regions on all devices both **−−allregions** and **−−alldevices** must be used.

If a **−−groupid** is given instead of a **−−regionid** the command will attempt to delete the group and all regions that it contains.

If a deleted region is the first member of a group of regions the group will also be removed.

group [*device_name*|**−−alldevices**] [**−−alias** *name*] [**−−regions** *regions*]
Combine one or more statistics regions on the specified device into a group.

The list of regions to be grouped is specified with **−−regions** and an optional alias may be assigned with **−−alias**. The set of regions is given as a comma-separated list of region identifiers. A continuous range of identifers spanning from **R1** to **R2** may be expressed as '**R1−R2**'.

Regions that have a histogram configured can be grouped: in this case the number of histogram bins and their bounds must match exactly.

On success the group list and newly created **group_id** are printed to stdout.

The group metadata is stored with the first (lowest numbered) **region_id** in the group: deleting this

region will also delete the group and other group members will be returned to their prior state.

help [**−c**|**−C**|**−−columns**]
Outputs a summary of the commands available, optionally including the list of report fields.

list [*device_name*] [**−−histogram**] [**−−allprograms**|**−−programid** *id*] [**−−units** *units*] [**−−area**] [**−−region**] [**−−group**] [**−−nosuffix**] [**−−notimesuffix**] [**−v**|**−−verbose]**
List the statistics regions, areas, or groups registered on the device. If the **−−allprograms** switch is given all regions will be listed regardless of region program ID values.

By default only regions and groups are included in list output. If **−v** or **−−verbose** is given the report will also include a row of information for each configured group and for each area contained in each region displayed.

Regions that contain a single area are by default omitted from the verbose list since their properties are identical to the area that they contain − to view all regions regardless of the number of areas present use **−−region**). To also view the areas contained within regions use **−−area**.

If **−−histogram** is given the report will include the bin count and latency boundary values for any configured histograms.

print [*device_name*] [**−−clear**] [**−−allprograms**|**−−programid** *id*] [**−−allregions**|**−−regionid** *id*]
Print raw statistics counters for the specified region or for all present regions.

report [*device_name*] [**−−interval** *seconds*] [**−−count** *count*] [**−−units** *units*] [**−−histogram**] [**−−allprograms**|**−−programid** *id*] [**−−allregions**|**−−regionid** *id*] [**−−area**] [**−−region**] [**−−group**] [**−O**|**−−sort** *sort_fields*] [**−S**|**−−select** *selection*] [**−−units** *units*] [**−−nosuffix**] [**−−notimesuffix**]
Start a report for the specified object or for all present objects. If the count argument is specified, the report will repeat at a fixed interval set by the **−−interval** option. The default interval is one second.

If the **−−allprograms** switch is given, all regions will be listed, regardless of region program ID values.

If the **−−histogram** is given the report will include the histogram values and latency boundaries.

If the **−−relative** is used the default histogram field displays bin values as a percentage of the total number of I/Os.

Object types (areas, regions and groups) to include in the report are selected using the **−−area**, **−−region**, and **−−group** options.

ungroup [*device_name*|**−−alldevices**] [**−−groupid** *id*]
Remove an existing group and return all the group's regions to their original state.

The group to be removed is specified using **−−groupid**.

update_filemap *file_path* [**−−groupid** *id*] [**−−follow** *follow_mode*] [**−−foreground**]
Update a group of **dmstats** regions specified by **group_id**, that were previously created with **−−filemap**, either directly, or by starting the monitoring daemon, **dmfilemapd**.

This will add and remove regions to reflect changes in the allocated extents of the file on-disk, since the time that it was crated or last updated.

Use of this command is not normally needed since the **dmfilemapd** daemon will automatically monitor filemap groups and perform these updates when required.

If a filemapped group was created with **−−nomonitor**, or the daemon has been killed, the

update_filemap can be used to manually force an update or start a new daemon.

Use **--nomonitor** to force a direct update and disable starting the monitoring daemon.

REGIONS, AREAS, AND GROUPS

The device-mapper statistics facility allows separate performance counters to be maintained for arbitrary regions of devices. A region may span any range: from a single sector to the whole device. A region may be further sub-divided into a number of distinct areas (one or more), each with its own counter set. In this case a summary value for the entire region is also available for use in reports.

In addition, one or more regions on one device can be combined into a statistics group. Groups allow several regions to be aggregated and reported as a single entity; counters for all regions and areas are summed and used to report totals for all group members. Groups also permit the assignment of an optional alias, allowing meaningful names to be associated with sets of regions.

The group metadata is stored with the first (lowest numbered) **region_id** in the group: deleting this region will also delete the group and other group members will be returned to their prior state.

By default new regions span the entire device. The **--start** and **--length** options allows a region of any size to be placed at any location on the device.

Using offsets it is possible to create regions that map individual objects within a block device (for example: partitions, files in a file system, or stripes or other structures in a RAID volume). Groups allow several non-contiguous regions to be assembled together for reporting and data aggregation.

A region may be either divided into the specified number of equal-sized areas, or into areas of the given size by specifying one of **--areas** or **--areasize** when creating a region with the **create** command. Depending on the size of the areas and the device region the final area within the region may be smaller than requested.

Region identifiers

Each region is assigned an identifier when it is created that is used to reference the region in subsequent operations. Region identifiers are unique within a given device (including across different **program_id** values).

Depending on the sequence of create and delete operations, gaps may exist in the sequence of **region_id** values for a particular device.

The **region_id** should be treated as an opaque identifier used to reference the region.

Group identifiers

Groups are also assigned an integer identifier at creation time; like region identifiers, group identifiers are unique within the containing device.

The **group_id** should be treated as an opaque identifier used to reference the group.

FILE MAPPING

Using **--filemap**, it is possible to create regions that correspond to the extents of a file in the file system. This allows IO statistics to be monitored on a per-file basis, for example to observe large database files, virtual machine images, or other files of interest.

To be able to use file mapping, the file must be backed by a device-mapper device, and in a file system that supports the FIEMAP ioctl (and which returns data describing the physical location of extents). This currently includes **xfs(5)** and **ext4(5)**.

By default the regions making up a file are placed together in a group, and the group alias is set to the

basename(3) of the file. This allows statistics to be reported for the file as a whole, aggregating values for the regions making up the group. To see only the whole file (group) when using the **list** and **report** commands, use **−−group**.

Since it is possible for the file to change after the initial group of regions is created, the **update_filemap** command, and **dmfilemapd** daemon are provided to update file mapped groups either manually or automatically.

File follow modes

The file map monitoring daemon can monitor files in two distinct ways: follow-inode mode, and follow-path mode.

The mode affects the behaviour of the daemon when a file under monitoring is renamed or unlinked, and the conditions which cause the daemon to terminate.

If follow-inode mode is used, the daemon will hold the file open, and continue to update regions from the same file descriptor. This means that the mapping will follow rename, move (within the same file system), and unlink operations. This mode is useful if the file is expected to be moved, renamed, or unlinked while it is being monitored.

In follow-inode mode, the daemon will exit once it detects that the file has been unlinked and it is the last holder of a reference to it.

If follow-path is used, the daemon will re-open the provided path on each monitoring iteration. This means that the group will be updated to reflect a new file being moved to the same path as the original file. This mode is useful for files that are expected to be updated via unlink and rename.

In follow-path mode, the daemon will exit if the file is removed and not replaced within a brief tolerance interval (one second).

To stop the daemon, delete the group containing the mapped regions: the daemon will automatically shut down.

The daemon can also be safely killed at any time and the group kept: if the file is still being allocated the mapping will become progressively out-of−date as extents are added and removed (in this case the daemon can be re-started or the group updated manually with the **update_filemap** command).

See the **create** command and **−−filemap**, **−−follow**, and **−−nomonitor** options for further information.

Limitations

The daemon attempts to maintain good synchronisation between the file extents and the regions contained in the group, however, since it can only react to new allocations once they have been written, there are inevitably some IO events that cannot be counted when a file is growing, particularly if the file is being extended by a single thread writing beyond end-of−file (for example, the **dd** program).

There is a further loss of events in that there is currently no way to atomically resize a **dmstats** region and preserve its current counter values. This affects files when they grow by extending the final extent, rather than allocating a new extent: any events that had accumulated in the region between any prior operation and the resize are lost.

File mapping is currently most effective in cases where the majority of IO does not trigger extent allocation. Future updates may address these limitations when kernel support is available.

REPORT FIELDS

The dmstats report provides several types of field that may be added to the default field set, or used to create custom reports.

All performance counters and metrics are calculated per-area.

Derived metrics

A number of metrics fields are included that provide high level performance indicators. These are based on the fields provided by the conventional Linux iostat program and are derived from the basic counter values provided by the kernel for each area.

reads_merged_per_sec
Reads merged per second.

writes_merged_per_sec
Writes merged per second.

reads_per_sec
Reads completed per second.

writes_per_sec
Writes completed per second.

read_size_per_sec
Size of data read per second.

write_size_per_sec
Size of data written per second.

avg_request_size
Average request size.

queue_size
Average queue size.

await The average wait time for read and write operations.

r_await
The average wait time for read operations.

w_await
The average wait time for write operations.

throughput
The device throughput in operations per second.

service_time
The average service time (in milliseconds) for operations issued to the device.

util Percentage of CPU time during which I/O requests were issued to the device (bandwidth utilization for the device). Device saturation occurs when this value is close to 100%.

Group, region and area meta fields

Meta fields provide information about the groups, regions, or areas that the statistics values relate to. This includes the region and area identifier, start, length, and counts, as well as the program ID and user data values.

region_id
Region identifier. This is a non-negative integer returned by the kernel when a statistics region is created.

region_start
The region start location. Display units are selected by the **--units** option.

region_len
The length of the region. Display units are selected by the **--units** option.

area_id
Area identifier. Area identifiers are assigned by the device-mapper statistics library and uniquely identify each area within a region. Each ID corresponds to a distinct set of performance counters for that area of the statistics region. Area identifiers are always monotonically increasing within a region so that higher ID values correspond to greater sector addresses within the area and no gaps in the sequence of identifiers exist.

area_start
The area start location. Display units are selected by the **--units** option.

area_len
The length of the area. Display units are selected by the **--units** option.

area_count
The number of areas in this region.

program_id
The program ID value associated with this region.

user_data
The user data value associated with this region.

group_id
Group identifier. This is a non-negative integer returned by the dmstats **group** command when a statistics group is created.

interval_ns
The estimated interval over which the current counter values have accumulated. The value is reported as an interger expressed in units of nanoseconds.

interval
The estimated interval over which the current counter values have accumulated. The value is reported as a real number in units of seconds.

Basic counters
Basic counters provide access to the raw counter data from the kernel, allowing further processing to be carried out by another program.

The kernel provides thirteen separate counters for each statistics area. The first eleven of these match the counters provided in /proc/diskstats or /sys/block/*/*/stat. The final pair provide separate counters for read and write time.

read_count
Count of reads completed this interval.

reads_merged_count
Count of reads merged this interval.

read_sector_count
Count of 512 byte sectors read this interval.

read_time
Accumulated duration of all read requests (ns).

write_count
Count of writes completed this interval.

writes_merged_count
Count of writes merged this interval.

write_sector_count
Count of 512 byte sectors written this interval.

write_time
Accumulated duration of all write requests (ns).

in_progress_count
Count of requests currently in progress.

io_ticks
Nanoseconds spent servicing requests.

queue_ticks
This field is incremented at each I/O start, I/O completion, I/O merge, or read of these stats by the number of I/Os in progress multiplied by the number of milliseconds spent doing I/O since the last update of this field. This can provide an easy measure of both I/O completion time and the backlog that may be accumulating.

read_ticks
Nanoseconds spent servicing reads.

write_ticks
Nanoseconds spent servicing writes.

Histogram fields
Histograms measure the frequency distribution of user specified I/O latency intervals. Histogram bin boundaries are specified when a region is created.

A brief representation of the histogram values and latency intervals can be included in the report using these fields.

hist_count
A list of the histogram counts for the current statistics area in order of ascending latency value. Each value represents the number of I/Os with latency times falling into that bin's time range during the sample period.

hist_count_bounds
A list of the histogram counts for the current statistics area in order of ascending latency value including bin boundaries: each count is prefixed by the lower bound of the corresponding histogram bin.

hist_count_ranges
A list of the histogram counts for the current statistics area in order of ascending latency value including bin boundaries: each count is prefixed by both the lower and upper bounds of the corresponding histogram bin.

hist_percent
A list of the relative histogram values for the current statistics area in order of ascending latency value, expressed as a percentage. Each value represents the proportion of I/Os with latency times falling into that bin's time range during the sample period.

hist_percent_bounds
A list of the relative histogram values for the current statistics area in order of ascending latency value, expressed as a percentage and including bin boundaries. Each value represents the proportion of I/Os with latency times falling into that bin's time range during the sample period and is prefixed with the corresponding bin's lower bound.

hist_percent_ranges
A list of the relative histogram values for the current statistics area in order of ascending latency value, expressed as a percentage and including bin boundaries. Each value represents the proportion of I/Os with latency times falling into that bin's time range during the sample period and is prefixed with the corresponding bin's lower and upper bounds.

hist_bounds
A list of the histogram boundary values for the current statistics area in order of ascending latency value. The values are expressed in whole units of seconds, miliseconds, microseconds or nanoseconds with a suffix indicating the unit.

hist_ranges
A list of the histogram bin ranges for the current statistics area in order of ascending latency value. The values are expressed as "LOWER-UPPER" in whole units of seconds, miliseconds, microseconds or nanoseconds with a suffix indicating the unit.

hist_bins
The number of latency histogram bins configured for the area.

EXAMPLES

Create a whole-device region with one area on vg00/lvol1
 # dmstats create vg00/lvol1
vg00/lvol1: Created new region with 1 area(s) as region ID 0

Create a 32M region 1G into device d0
 # dmstats create −−start 1G −−length 32M d0
d0: Created new region with 1 area(s) as region ID 0

Create a whole-device region with 8 areas on every device
 # dmstats create −−areas 8
vg00−lvol1: Created new region with 8 area(s) as region ID 0
vg00−lvol2: Created new region with 8 area(s) as region ID 0
vg00−lvol3: Created new region with 8 area(s) as region ID 0
vg01−lvol0: Created new region with 8 area(s) as region ID 2
vg01−lvol1: Created new region with 8 area(s) as region ID 0
vg00−lvol2: Created new region with 8 area(s) as region ID 1

Delete all regions on all devices
 # dmstats delete −−alldevices −−allregions

Create a whole-device region with areas 10GiB in size on vg00/lvol1 using dmsetup
 # dmsetup stats create −−areasize 10G vg00/lvol1
vg00−lvol1: Created new region with 5 area(s) as region ID 1

Create a 1GiB region with 16 areas at the start of vg00/lvol1
 # dmstats create −−start 0 −−len 1G −−areas=16 vg00/lvol1
vg00−lvol1: Created new region with 16 area(s) as region ID 0

List the statistics regions registered on vg00/lvol1
 # dmstats list vg00/lvol1

Name	RgID	RStart	RSize	#Areas	ASize	ProgID
vg00−lvol1	0	0	61.00g	1	61.00g	dmstats
vg00−lvol1	1	61.00g	19.20g	1	19.20g	dmstats
vg00−lvol1	2	80.20g	2.14g	1	2.14g	dmstats

Display five statistics reports for vg00/lvol1 at an interval of one second
 # dmstats report −−interval 1 −−count 5 vg00/lvol1
 # dmstats report

Name	RgID	ArID	AStart	ASize	RRqM/s	WRqM/s	R/s	W/s	RSz/s	WSz/s	AvRqSz	QSize	Util%	AWait	RdAWa	WrAWa
vg_hex−lv_home	0	0	0	61.00g	0.00	0.00	0.00	218.00	0	1.04m	4.50k	2.97	81.70	13.62	0.00	13.62
vg_hex−lv_home	1	0	61.00g	19.20g	0.00	0.00	0.00	5.00	0	548.00k	109.50k	0.14	11.00	27.40	0.00	27.40
vg_hex−lv_home	2	0	80.20g	2.14g	0.00	0.00	0.00	14.00	0	1.15m	84.00k	0.39	18.70	27.71	0.00	27.71

Create one region for reach target contained in device vg00/lvol1
 # dmstats create −−segments vg00/lvol1
vg00−lvol1: Created new region with 1 area(s) as region ID 0
vg00−lvol1: Created new region with 1 area(s) as region ID 1
vg00−lvol1: Created new region with 1 area(s) as region ID 2

Create regions mapping each file in the directory images/ and place them into separate groups, each named after the corresponding file
 # dmstats create −−filemap images/*
images/vm1.qcow2: Created new group with 87 region(s) as group ID 0.
images/vm1−1.qcow2: Created new group with 8 region(s) as group ID 87.
images/vm2.qcow2: Created new group with 11 region(s) as group ID 95.
images/vm2−1.qcow2: Created new group with 1454 region(s) as group ID 106.
images/vm3.img: Created new group with 2 region(s) as group ID 1560.

Print raw counters for region 4 on device d0
 # dmstats print −−regionid 4 d0
2097152+65536 0 0 0 0 29 0 264 701 0 41 701 0 41

AUTHORS

Bryn M. Reeves <bmr@redhat.com>

SEE ALSO

dmsetup(8)

LVM2 resource page: https://www.sourceware.org/lvm2/
Device-mapper resource page: http://sources.redhat.com/dm/

Device-mapper statistics kernel documentation
Documentation/device-mapper/statistics.txt