Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'blkparse.1' command

$ man blkparse.1

BLKPARSE(1) BLKPARSE(1)

NAME

blkparse - produce formatted output of event streams of block devices

SYNOPSIS

blkparse [options]

DESCRIPTION

The blkparse utility will attempt to combine streams of events for var? ious devices on various CPUs, and produce a formatted output of the event information. Specifically, it will take the (machine-readable) output of the blktrace utility and convert it to a nicely formatted and human-readable form.

As with blktrace, some details concerning blkparse will help in under? standing the command line options presented below.

- By default, blkparse expects to run in a post-processing mode; one where the trace events have been saved by a previous run of blktrace, and blkparse is combining event streams and dumping formatted data. blkparse may be run in a live manner concurrently with blktrace by specifying -i - to blkparse, and combining it with the live option for blktrace. An example would be:

  % blktrace -d /dev/sda -o - | blkparse -i -

- You can set how many blkparse batches event reads via the -b option, the default is to handle events in batches of 512.

- If you have saved event traces in blktrace with different output

names (via the -o option to blktrace), you must specify the same in? put name via the -i option.

- The format of the output data can be controlled via the -f or -F op? tions -- see OUTPUT DESCRIPTION AND FORMATTING for details.

By default, blkparse sends formatted data to standard output. This may be changed via the -o option, or text output can be disabled via the -O option. A merged binary stream can be produced using the -d option.

OPTIONS

-A hex-mask

--set-mask=hex-mask

Set filter mask to hex-mask, see blktrace (8) for masks

-a mask

--act-mask=mask

Add mask to current filter, see blktrace (8) for masks

-D dir

--input-directory=dir

Prepend dir to input file names

-b batch

--batch={batch}

Standard input read batching

-i file

--input=file

Specifies base name for input files -- default is device.blk? trace.cpu.

As noted above, specifying -i - runs in live mode with blktrace (reading data from standard in).

-F typ,fmt

--format=typ,fmt

-f fmt

--format-spec=fmt

Sets output format (See OUTPUT DESCRIPTION AND FORMATTING for details.)

The -f form specifies a format for all events

The -F form allows one to specify a format for a specific event type. The single-character typ field is one of the action speci? fiers described in ACTION IDENTIFIERS.

-M

--no-msgs

When -d is specified, this will stop messages from being output to the file. (Can seriously reduce the size of the resultant file when using the CFQ I/O scheduler.)

-h

--hash-by-name

Hash processes by name, not by PID

-o file

--output=file

Output file

-O

--no-text-output

Do not produce text output, used for binary (-d) only

-d file

--dump-binary=file

Binary output file

-q

--quiet

Quiet mode

-s

--per-program-stats

Displays data sorted by program

-t

--track-ios

Display time deltas per IO

-w span

--stopwatch=span

Display traces for the span specified -- where span can be:

end-time -- Display traces from time 0 through end-time (in ns)

or

start:end-time -- Display traces from time start through end-time (in ns).

-v

--verbose

More verbose marginal on marginal errors

-V

--version

Display version

TRACE ACTIONS

The following trace actions are recognised:

C -- complete A previously issued request has been completed. The out? put will detail the sector and size of that request, as well as the success or failure of it.

D -- issued A request that previously resided on the block layer queue or in the i/o scheduler has been sent to the driver.

I -- inserted A request is being sent to the i/o scheduler for addition to the internal queue and later service by the driver. The request is fully formed at this time.

Q -- queued This notes intent to queue i/o at the given location. No real requests exists yet.

B -- bounced The data pages attached to this bio are not reachable by the hardware and must be bounced to a lower memory location. This causes a big slowdown in i/o performance, since the data must be copied to/from kernel buffers. Usually this can be fixed with using better hardware -- either a better i/o controller, or a platform with an IOMMU.

M -- back merge A previously inserted request exists that ends on the boundary of where this i/o begins, so the i/o scheduler can merge them together.

F -- front merge Same as the back merge, except this i/o ends where a previously inserted requests starts.

M --front or back merge One of the above

M -- front or back merge One of the above.

G -- get request To send any type of request to a block device, a struct request container must be allocated first.

S -- sleep No available request structures were available, so the is? suer has to wait for one to be freed.

P -- plug When i/o is queued to a previously empty block device queue, Linux will plug the queue in anticipation of future ios being added before this data is needed.

U -- unplug Some request data already queued in the device, start send? ing requests to the driver. This may happen automatically if a timeout period has passed (see next entry) or if a number of re? quests have been added to the queue.

T -- unplug due to timer If nobody requests the i/o that was queued af? ter plugging the queue, Linux will automatically unplug it after a defined period has passed.

X -- split On raid or device mapper setups, an incoming i/o may strad? dle a device or internal zone and needs to be chopped up into smaller pieces for service. This may indicate a performance problem due to a bad setup of that raid/dm device, but may also just be part of normal boundary conditions. dm is notably bad at this and will clone lots of i/o.

A -- remap For stacked devices, incoming i/o is remapped to device be? low it in the i/o stack. The remap action details what exactly is being remapped to what.

OUTPUT DESCRIPTION AND FORMATTING

The output from blkparse can be tailored for specific use -- in partic? ular, to ease parsing of output, and/or limit output fields to those the user wants to see. The data for fields which can be output include:

a Action, a (small) string (1 or 2 characters) -- see table below for more details

c CPU id

C Command

d RWBS field, a (small) string (1-3 characters) -- see section below

for more details

D 7-character string containing the major and minor numbers of the event's device (separated by a comma).

e Error value

m Minor number of event's device.

M Major number of event's device.

n Number of blocks

N Number of bytes

p Process ID

P Display packet data -- series of hexadecimal values

s Sequence numbers

S Sector number

t Time stamp (nanoseconds)

T Time stamp (seconds)

u Elapsed value in microseconds (-t command line option)

U Payload unsigned integer

Note that the user can optionally specify field display width, and op? tionally a left-aligned specifier. These precede field specifiers, with a '%' character, followed by the optional left-alignment specifier (-) followed by the width (a decimal number) and then the field.

Thus, to specify the command in a 12-character field that is left aligned:

-f "%-12C"

ACTION IDENTIFIERS

The following table shows the various actions which may be output:

A IO was remapped to a different device

B IO bounced

C IO completion

D IO issued to driver

F IO front merged with request on queue

G Get request

I IO inserted onto request queue

M IO back merged with request on queue

P Plug request

Q IO handled by request queue code

S Sleep request

T Unplug due to timeout

U Unplug request

X Split

RWBS DESCRIPTION

This is a small string containing at least one character ('R' for read, 'W' for write, or 'D' for block discard operation), and optionally ei? ther a 'B' (for barrier operations) or 'S' (for synchronous opera? tions).

DEFAULT OUTPUT

The standard header (or initial fields displayed) include:

"%D %2c %8s %5T.%9t %5p %2a %3d"

Breaking this down:

%D Displays the event's device major/minor as: %3d,%-3d.

%2c CPU ID (2-character field).

%8s Sequence number

%5T.%9t

5-character field for the seconds portion of the time stamp and a 9-character field for the nanoseconds in the time stamp.

%5p 5-character field for the process ID.

%2a 2-character field for one of the actions.

%3d 3-character field for the RWBS data.

Seeing this in action:

8,0 3 1 0.000000000 697 G W 223490 + 8 [kjournald]

The header is the data in this line up to the 223490 (starting block). The default output for all event types includes this header.

DEFAULT OUTPUT PER ACTION

C -- complete

If a payload is present, this is presented between parenthesis fol?

lowing the header, followed by the error value.

If no payload is present, the sector and number of blocks are pre? sented (with an intervening plus (+) character). If the -t option was specified, then the elapsed time is presented. In either case, it is followed by the error value for the completion.

B -- bounced

D -- issued

I -- inserted

Q -- queued

If a payload is present, the number of payload bytes is output, followed by the payload in hexadecimal between parenthesis.

If no payload is present, the sector and number of blocks are pre? sented (with an intervening plus (+) character). If the -t option was specified, then the elapsed time is presented (in parenthesis). In either case, it is followed by the command associated with the event (surrounded by square brackets).

F -- front merge

G -- get request

M -- back merge

S -- sleep

The starting sector and number of blocks is output (with an inter? vening plus (+) character), followed by the command associated with the event (surrounded by square brackets).

P -- plug

The command associated with the event (surrounded by square brack? ets) is output.

U -- unplug

T -- unplug due to timer

The command associated with the event (surrounded by square brack? ets) is output, followed by the number of requests outstanding.

X -- split

The original starting sector followed by the new sector (separated by a slash (/) is output, followed by the command associated with

the event (surrounded by square brackets).

A -- remap

Sector and length is output, along with the original device and sector offset.

EXAMPLES

To trace the i/o on the device /dev/sda and parse the output to human readable form, use the following command:

```
% blktrace -d /dev/sda -o - | blkparse -i -
```

(see blktrace (8) for more information). This same behaviour can be achieve with the convenience script btrace. The command

```
% btrace /dev/sda
```

has exactly the same effect as the previous command. See btrace (8) for more information.

To trace the i/o on a device and save the output for later processing with blkparse, use blktrace like this:

```
% blktrace /dev/sda /dev/sdb
```

This will trace i/o on the devices /dev/sda and /dev/sdb and save the recorded information in the files sda and sdb in the current directory, for the two different devices, respectively. This trace information can later be parsed by the blkparse utility:

```
% blkparse sda sdb
```

which will output the previously recorded tracing information in human readable form to stdout.

AUTHORS

blkparse was written by Jens Axboe, Alan D. Brunelle and Nathan Scott. This man page was created from the blktrace documentation by Bas Zoetekouw.

REPORTING BUGS

Report bugs to <linux-btrace@vger.kernel.org>

COPYRIGHT

censes/gpl.html>. There is NO WARRANTY, to the extent permitted by law.

This manual page was created for Debian by Bas Zoetekouw. It was de? rived from the documentation provided by the authors and it may be used, distributed and modified under the terms of the GNU General Pub? lic License, version 2.

On Debian systems, the text of the GNU General Public License can be found in /usr/share/common-licenses/GPL-2.

SEE ALSO

btrace (8), blktrace (8), verify_blkparse (1), blkrawverify (1), btt (1)