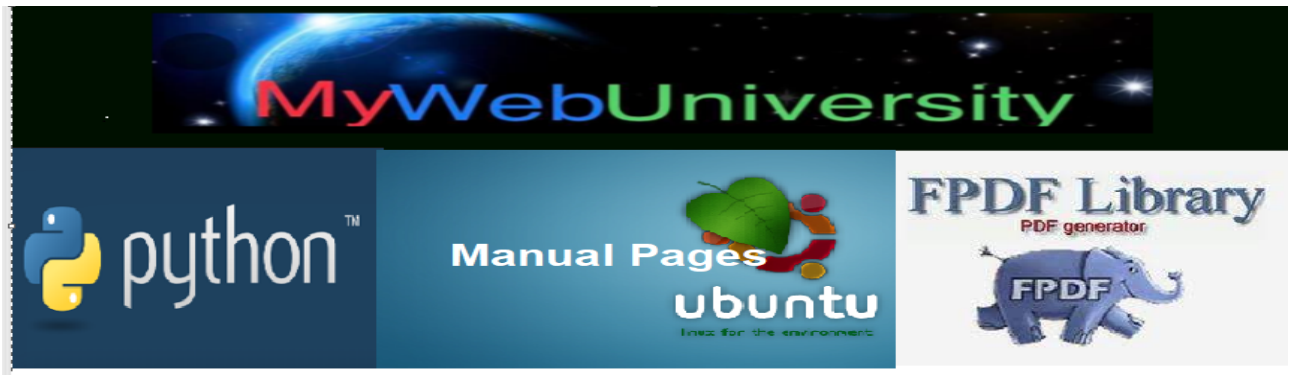

Full credit is given to the above companies including the OS that this PDF file was generated!

Linux Ubuntu 26.04 Manual Pages on command 'trace-cmd-set.1'

$ man trace-cmd-set.1

TRACE-CMD-SET(1) libtracefs Manual TRACE-CMD-SET(1)

NAME

trace-cmd-set - set a configuration parameter of the Ftrace Linux internal tracer

SYNOPSIS

trace-cmd set [OPTIONS] [command]

DESCRIPTION

The trace-cmd(1) set command will set a configuration parameter of the Ftrace Linux kernel tracer. The specified command will be run after the ftrace state is set. The configured ftrace state can be restored to default using the trace-cmd-reset(1) command.

OPTIONS

-p tracer

Specify a tracer. Tracers usually do more than just trace an event. Common tracers are: function, function_graph, preemptirqsoff, irqsoff, preemptoff and wakeup. A tracer must be supported by the running kernel. To see a list of available tracers, see trace-cmd-list(1).

-e event

Specify an event to trace. Various static trace points have been added to the Linux kernel. They are grouped by subsystem where you can enable all events of a given subsystem or specify specific events to be enabled. The event is of the format "subsystem:event-name". You can also just specify the subsystem without the :event-name or the event-name without the "subsystem:". Using "-e sched_switch" will enable the "sched_switch" event where as, "-e sched" will enable all events under the "sched" subsystem.

The 'event' can also contain glob expressions. That is, "*stat*" will select all events (or subsystems) that have the characters "stat" in their names.

The keyword 'all' can be used to enable all events.

-T

Enable a stacktrace on each event. For example:

```
<idle>-0     [003] 58549.289091: sched_switch:         kworker/0:1:0 [120] R ==> trace-cmd:2603 [120]
<idle>-0     [003] 58549.289092: kernel_stack:         <stack trace>
=> schedule (ffffffff814b260e)
=> cpu_idle (ffffffff8100a38c)
=> start_secondary (ffffffff814ab828)
```

--func-stack

Enable a stack trace on all functions. Note this is only applicable for the "function" plugin tracer, and will only take effect if the -l option is used and succeeds in limiting functions. If the function tracer is not filtered, and the stack trace is enabled, you can live lock the machine.

-f filter

Specify a filter for the previous event. This must come after a -e. This will filter what events get recorded based on the content of the event. Filtering is passed to the kernel directly so what filtering is allowed may depend on what version of the kernel you have. Basically, it will let you use C notation to check if an event should be processed or not.

==, >=, <=, >, <, &, |, && and ||

The above are usually safe to use to compare fields.

-R trigger

Specify a trigger for the previous event. This must come after a -e. This will add a given trigger to the given event. To only enable the trigger and not the event itself, then place the event after the -v option.

See Documentation/trace/events.txt in the Linux kernel source for more information on triggers.

-v

This will negate options specified after it on the command line. It affects:

-e: Causes all specified events to not be traced. This is useful for

selecting a subsystem to be traced but to leave out various events.

For example: "-e sched -v -e "*stat*"" will enable all events in

the sched subsystem except those that have "stat" in their names.

-B: Deletes the specified ftrace instance. There must be no

configuration options related to this instance in the command line.

For example: "-v -B bar -B foo" will delete instance bar and create

a new instance foo.

Note: the -v option was taken from the way grep(1) inverts the following

matches.

-P pid

This will filter only the specified process IDs. Using -P will let you trace only events

that are caused by the process.

-c

Used -P to trace the process' children too (if kernel supports it).

--user

Execute the specified command as given user.

-C clock

Set the trace clock to "clock".

Use trace-cmd(1) list -C to see what clocks are available.

-l function-name

This will limit the function and function_graph tracers to only trace the given function

name. More than one -l may be specified on the command line to trace more than one

function. The limited use of glob expressions are also allowed. These are match* to only

filter functions that start with match. *match to only filter functions that end with

match. *match* to only filter on functions that contain match.

-g function-name

This option is for the function_graph plugin. It will graph the given function. That is,

it will only trace the function and all functions that it calls. You can have more than

one -g on the command line.

-n function-name

This has the opposite effect of -l. The function given with the -n option will not be

traced. This takes precedence, that is, if you include the same function for both -n and

-l, it will not be traced.

-d

Some tracer plugins enable the function tracer by default. Like the latency tracers.

This option prevents the function tracer from being enabled at start up.

-D

The option -d will try to use the function-trace option to disable the function tracer (if available), otherwise it defaults to the proc file: /proc/sys/kernel/ftrace_enabled, but will not touch it if the function-trace option is available. The -D option will disable both the ftrace_enabled proc file as well as the function-trace option if it exists.

Note, this disable function tracing for all users, which includes users outside of ftrace tracers (stack_tracer, perf, etc).

-O option

Ftrace has various options that can be enabled or disabled. This allows you to set them. Appending the text no to an option disables it. For example: "-O nograph-time" will disable the "graph-time" Ftrace option.

-b size

This sets the ring buffer size to size kilobytes. Because the Ftrace ring buffer is per CPU, this size is the size of each per CPU ring buffer inside the kernel. Using "-b 10000" on a machine with 4 CPUs will make Ftrace have a total buffer size of 40 Megs.

-B buffer-name

If the kernel supports multiple buffers, this will add a buffer with the given name. If the buffer name already exists, that buffer is just reset.

After a buffer name is stated, all events added after that will be associated with that buffer. If no buffer is specified, or an event is specified before a buffer name, it will be associated with the main (toplevel) buffer.

trace-cmd set -e sched -B block -e block -B time -e timer sleep 1

The above is will enable all sched events in the main buffer. It will then create a 'block' buffer instance and enable all block events within that buffer. A 'time' buffer instance is created and all timer events will be enabled for that event.

-m size

The max size in kilobytes that a per cpu buffer should be. Note, due to rounding to page

size, the number may not be totally correct. Also, this is performed by switching between two buffers that are half the given size thus the output may not be of the given size even if much more was written.

Use this to prevent running out of diskspace for long runs.

-M cpumask

Set the cpumask for to trace. It only affects the last buffer instance given. If supplied before any buffer instance, then it affects the main buffer. The value supplied must be a hex number.

trace-cmd set -p function -M c -B events13 -e all -M 5

If the -M is left out, then the mask stays the same. To enable all CPUs, pass in a value of '-1'.

-i

By default, if an event is listed that trace-cmd does not find, it will exit with an error. This option will just ignore events that are listed on the command line but are not found on the system.

-q | --quiet

Suppresses normal output, except for errors.

--max-graph-depth depth

Set the maximum depth the function_graph tracer will trace into a function. A value of one will only show where userspace enters the kernel but not any functions called in the kernel. The default is zero, which means no limit.

--cmdlines-size size

Set the number of entries the kernel tracing file "saved_cmdlines" can contain. This file is a circular buffer which stores the mapping between cmdlines and PIDs. If full, it leads to unresolved cmdlines ("<...>") within the trace. The kernel default value is 128.

--module module

Filter a module?s name in function tracing. It is equivalent to adding :mod:module after all other functions being filtered. If no other function filter is listed, then all modules functions will be filtered in the filter.

'--module snd' is equivalent to '-l :mod:snd'

'--module snd -l "*jack*"' is equivalent to '-l "*jack*:mod:snd"'

'--module snd -n "*"' is equivalent to '-n :mod:snd'

--stderr

Have output go to stderr instead of stdout, but the output of the command executed will not be changed. This is useful if you want to monitor the output of the command being executed, but not see the output from trace-cmd.

--fork

If a command is listed, then trace-cmd will wait for that command to finish, unless the --fork option is specified. Then it will fork the command and return immediately.

--verbose[=level]

Set the log level. Supported log levels are "none", "critical", "error", "warning", "info", "debug", "all" or their identifiers "0", "1", "2", "3", "4", "5", "6". Setting the log level to specific value enables all logs from that and all previous levels. The level will default to "info" if one is not specified.

Example: enable all critical, error and warning logs

trace-cmd set --verbose=warning

EXAMPLES

Enable all events for tracing:

trace-cmd set -e all

Set the function tracer:

trace-cmd set -p function

SEE ALSO

trace-cmd(1), trace-cmd-report(1), trace-cmd-start(1), trace-cmd-stop(1), trace-cmd-extract(1), trace-cmd-reset(1), trace-cmd-split(1), trace-cmd-list(1), trace-cmd-listen(1), trace-cmd-profile(1)

AUTHOR

Written by Tzvetomir Stoyanov (VMware) <tz.stoyanov@gmail.com[1]>

RESOURCES

https://git.kernel.org/pub/scm/utils/trace-cmd/trace-cmd.git/

COPYING

NOTES

1. tz.stoyanov@gmail.com

mailto:tz.stoyanov@gmail.com