



Linux Ubuntu 26.04 Manual Pages on command 'tracecmd_read_cpu_first.3'

\$ man tracecmd_read_cpu_first.3

LIBTRACECMD(3) libtracefs Manual LIBTRACECMD(3)

NAME

tracecmd_read_cpu_first, tracecmd_read_data, tracecmd_read_at, tracecmd_free_record,
tracecmd_get_tep - Read recorded events from a trace file.

SYNOPSIS

```
#include <trace-cmd.h>

struct tep_record *tracecmd_read_cpu_first(struct tracecmd_input *handle, int cpu);
struct tep_record *tracecmd_read_data(struct tracecmd_input *handle, int cpu);
struct tep_record *tracecmd_read_at(struct tracecmd_input *handle, unsigned long long offset, int *cpu);
void tracecmd_free_record(struct tep_record *record);
struct tep_handle *tracecmd_get_tep(struct tracecmd_input *handle);
```

DESCRIPTION

This set of APIs can be used to read tracing data from a trace file opened with
tracecmd_open()(3), tracecmd_open_fd()(3) or tracecmd_open_head()(3).

The tracecmd_read_cpu_first() function reads the first trace record for a given cpu from a
trace file associated with handle. The returned record must be freed with
tracecmd_free_record().

The `tracecmd_read_data()` function reads the next trace record for a given cpu from a trace file associated with handle and increments the read location pointer, so that the next call to `tracecmd_read_data()` will not read the same record again. The returned record must be freed with `tracecmd_free_record()`.

The `tracecmd_read_at()` function reads a trace record from a specific offset within the file associated with handle. The CPU on which the recorded event occurred is stored in the cpu. The function does not change the current read location pointer. The returned record must be freed with `tracecmd_free_record()`.

The `tracecmd_free_record()` function frees a record returned by any of the `tracecmd_read_` APIs.

The `tracecmd_get_tep()` function returns a tep context for a given handle.

RETURN VALUE

The `tracecmd_read_cpu_first()`, `tracecmd_read_data()` and `tracecmd_read_at()` functions return a pointer to struct `tep_record` or NULL in case of an error. The returned record must be freed with `tracecmd_free_record()`.

The `tracecmd_get_tep()` function returns a pointer to tep context or NULL if there is no tep context for the given handle. The returned tep pointer must not be freed.

EXAMPLE

```
#include <trace-cmd.h>

...

struct tracecmd_input *handle = tracecmd_open("trace.dat");

if (!handle) {
    /* Failed to open trace.dat file */
}

...

unsigned long long offset = 0;

struct tep_record *rec;
```

```

int cpu = 0;

rec = tracecmd_read_cpu_first(handle, cpu);

while (rec) {

    ...

    if ( /* some interesting record noticed */ ) {

        /* store the offset of the interesting record */

        offset = rec->offset;

    }

    ...

    tracecmd_free_record(rec);

    rec = tracecmd_read_data(handle, cpu);

}

...

if (offset) {

    rec = tracecmd_read_at(handle, offset, &cpu);

    if (rec) {

        /* Got record at offset on cpu */

        ...

        tracecmd_free_record(rec);

    }

}

...

tracecmd_close(hadle);

```

FILES

trace-cmd.h

Header file to include in order to have access to the library APIs.

-ltracecmd

Linker switch to add when building a program that uses the library.

SEE ALSO

libtracefs(3), libtraceevent(3), trace-cmd(1) trace-cmd.dat(5)

AUTHOR

Steven Rostedt <rostedt@goodmis.org[1]>

Tzvetomir Stoyanov <tz.stoyanov@gmail.com[2]>

REPORTING BUGS

Report bugs to <linux-trace-devel@vger.kernel.org[3]>

LICENSE

libtracecmd is Free Software licensed under the GNU LGPL 2.1

RESOURCES

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

COPYING

Copyright (C) 2020 VMware, Inc. Free use of this software is granted under the terms of the GNU Public License (GPL).

NOTES

1. rostedt@goodmis.org

mailto:rostedt@goodmis.org

2. tz.stoyanov@gmail.com

mailto:tz.stoyanov@gmail.com

3. linux-trace-devel@vger.kernel.org

mailto:linux-trace-devel@vger.kernel.org