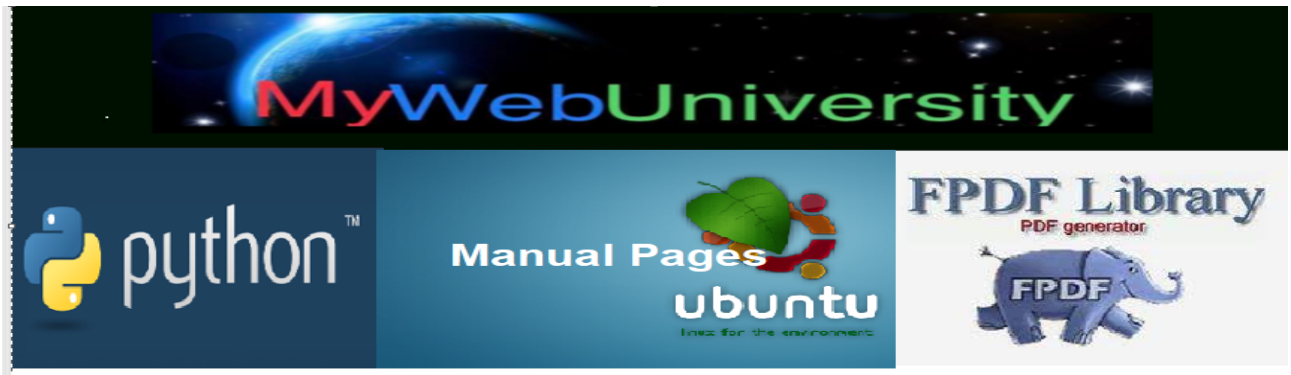

Full credit is given to the above companies including the OS that this PDF file was generated!

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

$ man tracecmd_add_ts_offset.3

LIBTRACECMD(3) libtracefs Manual LIBTRACECMD(3)

NAME

tracecmd_get_first_ts, tracecmd_add_ts_offset, tracecmd_get_tsc2nsec - Handle time stamps from a trace file.

SYNOPSIS

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

unsigned long long tracecmd_get_first_ts(struct tracecmd_input *handle);
void tracecmd_add_ts_offset(struct tracecmd_input *handle, long long offset);
int tracecmd_get_tsc2nsec(struct tracecmd_input *handle, int *mult, int pass[*]shift, unsigned long long *offset);
```

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_get_first_ts() function returns the time stamp of the first record in the handle.

The tracecmd_add_ts_offset() function adds an offset to each of the records in the handle that represents a trace file. This is useful for associating two different tracing files by their offset (for example a trace file from a host and a trace file from a guest that were

not synchronized when created).

The tracecmd_get_tsc2nsec returns the calculation values that convert the raw timestamp into nanoseconds. The parameters are pointers to the storage to save the values, or NULL to ignore them. The multiplier will be saved in mult, the shift value will be saved in shift, and the offset value will be saved in offset, if the corresponding parameters are not NULL.

RETURN VALUE

The tracecmd_get_first_ts() function returns the timestamp of the first record in a trace file for the given handle.

The tracecmd_get_tsc2nsec() returns 0 if the tracing clock supports the shift values and -1 otherwise. Note, that if the trace file has the TSC2NSEC option, the values returned in the parameters may still be valid even if the function itself returns -1. The return value only notes if the values will be used in the calculations of the given clock.

EXAMPLE

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

static int print_events(struct tracecmd_input *handle, struct tep_record *record, int cpu, void *data)
{
        static struct trace_seq seq;
        struct tep_handle *tep = tracecmd_get_tep(handle);
        const char *file = tracecmd_get_private(handle);

        if (!seq.buffer)
                trace_seq_init(&seq);

        trace_seq_reset(&seq);
        trace_seq_printf(&seq, "%s: ", file);
        tep_print_event(tep, &seq, record, "%6.1000d [%03d] %s-%d %s: %s\n",
                        TEP_PRINT_TIME, TEP_PRINT_CPU, TEP_PRINT_COMM, TEP_PRINT_PID,
```

```
                TEP_PRINT_NAME, TEP_PRINT_INFO);
        trace_seq_terminate(&seq);
        trace_seq_do_printf(&seq);
        return 0;
}

int main(int argc, char **argv)
{
        struct tracecmd_input **handles = NULL;
        unsigned long long ts, first_ts = 0;
        unsigned long long offset;
        int multi;
        int shift;
        int nr_handles = 0;
        int ret;
        int i;

        if (argc < 2) {
                printf("usage: %s trace.dat [trace.dat ...]\n", argv[0]);
                exit(-1);
        }

        for (i = 1; i < argc; i++) {
                handles = realloc(handles, sizeof(*handles) * (nr_handles + 1));
                if (!handles)
                        exit(-1);
                handles[nr_handles] = tracecmd_open(argv[i], 0);
                if (!handles[nr_handles])
                        exit(-1);

                ret = tracecmd_get_tsc2nsec(handles[nr_handles], &multi, &shift, &offset);
                if (!ret)
                        printf(" %s has tsc2nsec calculations of mult:%d shift:%d offset:%lld\n",
```

```
                        argv[i], multi, shift, offset);
                tracecmd_set_private(handles[nr_handles], argv[i]);
                ts = tracecmd_get_first_ts(handles[nr_handles]);
                if (!first_ts || ts < first_ts)
                        first_ts = ts;
                nr_handles++;
        }

        /* Set the time stamp to start at the first record found */
        for (i = 0; i < nr_handles; i++)
                tracecmd_add_ts_offset(handles[i], -first_ts);

        tracecmd_iterate_events_multi(handles, nr_handles, print_events, NULL);

        for (i = 0; i < nr_handles; i++)
                tracecmd_close(handles[i]);
        free(handles);
}
```

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

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