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 'posix_trace_getnext_event.3p' command

$ man posix_trace_getnext_event.3p

POSIX_TRACE_GETNEXT_EVENT(3POSIX Programmer's ManPOSIX_TRACE_GETNEXT_EVENT(3P)

PROLOG

This manual page is part of the POSIX Programmer's Manual. The Linux implementation of this interface may differ (consult the corresponding Linux manual page for details of Linux behavior), or the interface may not be implemented on Linux.

NAME

posix_trace_getnext_event, posix_trace_timedgetnext_event, posix_trace_trygetnext_event ? retrieve a trace event (TRACING)

SYNOPSIS

```
#include <sys/types.h>
#include <trace.h>
int posix_trace_getnext_event(trace_id_t trid,
    struct posix_trace_event_info *restrict event,
    void *restrict data, size_t num_bytes,
    size_t *restrict data_len, int *restrict unavailable);
int posix_trace_timedgetnext_event(trace_id_t trid,
    struct posix_trace_event_info *restrict event,
    void *restrict data, size_t num_bytes,
    size_t *restrict data_len, int *restrict unavailable,
    const struct timespec *restrict abstime);
int posix_trace_trygetnext_event(trace_id_t trid,
    struct posix_trace_event_info *restrict event,
```

void *restrict data, size_t num_bytes,

size_t *restrict data_len, int *restrict unavailable);

DESCRIPTION

The posix_trace_getnext_event() function shall report a recorded trace event either from an active trace stream without log or a pre-recorded trace stream identified by the trid argument. The posix_trace_tryget? next_event() function shall report a recorded trace event from an ac? tive trace stream without log identified by the trid argument.

The trace event information associated with the recorded trace event shall be copied by the function into the structure pointed to by the argument event and the data associated with the trace event shall be copied into the buffer pointed to by the data argument.

The posix_trace_getnext_event() function shall block if the trid argu? ment identifies an active trace stream and there is currently no trace event ready to be retrieved. When returning, if a recorded trace event was reported, the variable pointed to by the unavailable argument shall be set to zero. Otherwise, the variable pointed to by the unavailable argument shall be set to a value different from zero.

The posix_trace_timedgetnext_event() function shall attempt to get an? other trace event from an active trace stream without log, as in the posix_trace_getnext_event() function. However, if no trace event is available from the trace stream, the implied wait shall be terminated when the timeout specified by the argument abstime expires, and the function shall return the error [ETIMEDOUT].

The timeout shall expire when the absolute time specified by abstime passes, as measured by the clock upon which timeouts are based (that is, when the value of that clock equals or exceeds abstime), or if the absolute time specified by abstime has already passed at the time of the call.

The timeout shall be based on the CLOCK_REALTIME clock. The resolution of the timeout shall be the resolution of the clock on which it is based. The timespec data type is defined in the <time.h> header.

Under no circumstance shall the function fail with a timeout if a trace

event is immediately available from the trace stream. The validity of the abstime argument need not be checked if a trace event is immedi? ately available from the trace stream.

The behavior of this function for a pre-recorded trace stream is un? specified.

The posix_trace_trygetnext_event() function shall not block. This function shall return an error if the trid argument identifies a pre-recorded trace stream. If a recorded trace event was reported, the variable pointed to by the unavailable argument shall be set to zero. Otherwise, if no trace event was reported, the variable pointed to by the unavailable argument shall be set to a value different from zero.

The argument num_bytes shall be the size of the buffer pointed to by the data argument. The argument data_len reports to the application the length in bytes of the data record just transferred. If num_bytes is greater than or equal to the size of the data associated with the trace event pointed to by the event argument, all the recorded data shall be transferred. In this case, the truncation-status member of the trace event structure shall be either POSIX_TRACE_NOT_TRUNCATED, if the trace event data was recorded without truncation while tracing, or POSIX_TRACE_TRUNCATED_RECORD, if the trace event data was truncated when it was recorded. If the num_bytes argument is less than the length of recorded trace event data, the data transferred shall be truncated to a length of num_bytes, the value stored in the variable pointed to by data_len shall be equal to num_bytes, and the truncation-status mem? ber of the event structure argument shall be set to POSIX_TRACE_TRUN? CATED_READ (see the posix_trace_event_info structure defined in <trace.h>).

The report of a trace event shall be sequential starting from the old? est recorded trace event. Trace events shall be reported in the order in which they were generated, up to an implementation-defined time res? olution that causes the ordering of trace events occurring very close to each other to be unknown. Once reported, a trace event cannot be re? ported again from an active trace stream. Once a trace event is re?

ported from an active trace stream without log, the trace stream shall make the resources associated with that trace event available to record future generated trace events.

RETURN VALUE

Upon successful completion, these functions shall return a value of zero. Otherwise, they shall return the corresponding error number.

If successful, these functions store:

* The recorded trace event in the object pointed to by event
* The trace event information associated with the recorded trace event in the object pointed to by data
* The length of this trace event information in the object pointed to by data_len
* The value of zero in the object pointed to by unavailable

ERRORS

These functions shall fail if:

EINVAL The trace stream identifier argument trid is invalid.

The posix_trace_getnext_event() and posix_trace_timedgetnext_event() functions shall fail if:

EINTR The operation was interrupted by a signal, and so the call had no effect.

The posix_trace_trygetnext_event() function shall fail if:

EINVAL The trace stream identifier argument trid does not correspond to an active trace stream.

The posix_trace_timedgetnext_event() function shall fail if:

EINVAL There is no trace event immediately available from the trace stream, and the timeout argument is invalid.

ETIMEDOUT

No trace event was available from the trace stream before the specified timeout timeout expired.

The following sections are informative.

EXAMPLES

None.

APPLICATION USAGE

None.

RATIONALE

None.

FUTURE DIRECTIONS

These functions may be removed in a future version.

SEE ALSO

posix_trace_close(), posix_trace_create()

The Base Definitions volume of POSIX.1?2017, <sys_types.h>, <trace.h>

COPYRIGHT