NAME

pcap_get_required_select_timeout - get a timeout to be used when doing select() for a live capture

SYNOPSIS

#include <pcap/pcap.h>

const struct timeval *pcap_get_required_select_timeout(pcap_t *p);

DESCRIPTION

pcap_get_required_select_timeout() returns, on UNIX, a pointer to a **struct timeval** containing a value that must be used as the minimum timeout in **select**(2), **poll**(2), **epoll_wait**(2), and **kevent**(2) calls, or **NULL** if there is no such timeout. If a non-**NULL** value is returned, it must be used regardless of whether **pcap_get_selectable_fd**(3) returns **-1** for any descriptor on which those calls are being done. **pcap_get_required_select_timeout**() should be called for all **pcap_t**s before a call to **select**(), **poll**(), **epoll_wait**(), or **kevent**(), and any timeouts used for those calls should be updated as appropriate given the new value of the timeout.

For **kevent**(), one **EVFILT_TIMER** filter per selectable descriptor can be used, rather than using the timeout argument to **kevent**(); if the **EVFILT_TIMER** event for a particular selectable descriptor signals an event, **pcap_dispatch**(3) should be called for the corresponding **pcap_t**.

On Linux systems with **timerfd_create**(2), one timer object created by **timerfd_create**() per selectable descriptor can be used, rather than using the timeout argument to **epoll_wait**(); if the timer object for a particular selectable descriptor signals an event, **pcap_dispatch**(3) should be called for the corresponding **pcap_t**.

Otherwise, a timeout value no larger than the smallest of all timeouts returned by **pcap_get_required_select_timeout**() for devices from which packets will be captured and any other timeouts to be used in the call should be used as the timeout for the call, and, when the call returns, **pcap_dispatch**(3) should be called for all **pcap_t**s for which a non-**NULL** timeout was returned, regardless of whether it’s indicated as having anything to read from it or not.

All devices with a non-**NULL** timeout must be put in non-blocking mode with **pcap_setnonblock**(3).

Note that a device on which a read can be done without blocking may, on some platforms, not have any packets to read if the packet buffer timeout has expired. A call to **pcap_dispatch**() or **pcap_next_ex**(3) will return **0** in this case, but will not block.

pcap_get_required_select_timeout() is not available on Windows.

RETURN VALUE

A pointer to a **struct timeval** is returned if the timeout is required; otherwise **NULL** is returned.

BACKWARD COMPATIBILITY

This function became available in libpcap release 1.9.0. In previous releases, **select**(), **poll**(), **epoll_wait**(), and **kevent**() could not be used for devices that don't provide a selectable file descriptor (in other words, on any capture source for that **pcap_get_selectable_fd**() returns **-1**).

In libpcap release 1.10.0 and later, the timeout value can change from call to call, so **pcap_get_required_select_timeout**() must be called before each call to **select**(), **poll**(), **epoll_wait**(), or **kevent**(), and the new value must be used to calculate timeouts for the call. Code that does that will also work with libpcap 1.9.x releases, so code using **pcap_get_required_select_timeout**() should be changed to call it for each call to **select**(), **poll**(), **epoll_wait**(), or **kevent**() even if the code must also work with libpcap 1.9.x.

SEE ALSO

pcap(3), **pcap_get_selectable_fd**(3), **select**(2), **poll**(2), **epoll_wait**(2), **kqueue**(2)