ZebOS-XP BFD SMI Reference

IP Infusion Inc.

Generated by Doxygen 1.6.1

Wed Dec 16 12:33:43 2015

Contents

Chapter 1

File Index

1.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 2

File Documentation

2.1 smi_oam_bfd.h File Reference

Provides APIs for managing Bidirectional Forwarding Detection(BFD) in ZebOS.
`#include "smi_client.h"`

`#include "smi_oam_bfd_msg.h"`

Functions

- s_int32_t smi_bfd_notification_set (struct smiclient_globals ∗azg, u_int32_t vr_id, u_int32_t notif_flag)

  Sets to enable or disable the notification emission of BFD session on this device. If this object is set to true(1), then it enables the emission of bfdSessUp and bfdSessDown notifications; otherwise these notifications are not emitted.

- s_int32_t smi_bfd_get_sess_version_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗version, bool_t snmp)

  This function gets the version number of the BFD protocol.

- s_int32_t smi_bfd_get_sess_type_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗session_type, bool_t snmp)

  This function is used to get the session type for an SNMP Get request.

- s_int32_t smi_bfd_get_sess_mh_unlnk_mode_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗mh_unlnk_mode, bool_t snmp)

  This function is used to get the session multihop UNI(Unidirectional) link mode for SNMP Get request.

- s_int32_t smi_bfd_get_sess_disc_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_disc, bool_t snmp)

  This function is used to get the session discriminator for an SNMP Get request.

- s_int32_t smi_bfd_get_sess_rmte_disc_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_rmte_disc, bool_t snmp)

  This function is used to get the session discriminator chosen by the remote system for the BFD session.

- s_int32_t smi_bfd_get_sess_dest_udp_port_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_dest_udp_port, bool_t snmp)

  This function is used to get the destination UDP port number used for the BFD session's control packets.

- s_int32_t smi_bfd_get_sess_src_udp_port_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_src_udp_port, bool_t snmp)

  This function is used to get the source UDP port number used for the BFD session's control packets.

- s_int32_t smi_bfd_get_sess_echo_src_udp_port_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_echo_src_udp_port, bool_t snmp)

  This function is used to get the source UDP port number used for BFD session's echo packets.

- s_int32_t smi_bfd_get_sess_admin_status_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_admin_status, bool_t snmp)

  This function is used to get the session administration status.A transition from 'stop' to 'start' will start the BFD state machine for the session. The state machine will have an initial state of down.

- s_int32_t smi_bfd_get_sess_state_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_state, bool_t snmp)

  This function is used to get the session state.

- s_int32_t smi_bfd_get_sess_rmte_heard_flag_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_rmte_heard_flag, bool_t snmp)

  This function is used to get the status of BFD packet reception from the remote system. Specifically, it is set to true if the local system is actively receiving BFD packets from the remote system, and is set to false(2) if the local system has not received BFD packets recently.

- s_int32_t smi_bfd_api_get_sess_diag_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_diag, bool_t snmp)

  This function is used to get the session diagram, a diagnostic code specifying the local system's reason for the last transition of the session from up to some other state.

- s_int32_t smi_bfd_get_sess_oper_mode_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_oper_mode, bool_t snmp)

  This function is used to get the current operating mode that BFD session is operating in.

- s_int32_t smi_bfd_get_sess_dmnd_mode_dsrd_flag_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_dmnd_mode_dsrd_flag, bool_t snmp)

  This function is used to get the session demand mode desired flag which indicates that the local system's desire to use Demand mode.

- s_int32_t smi_bfd_get_sess_cntrlplane_indep_flag_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_cntrlplane_indep_flag, bool_t snmp)

  This function is used to get the session control plane independent flag indicates that the local system's ability to continue to function through a disruption of the control plane.

- s_int32_t smi_bfd_get_sess_interface_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_interface, bool_t snmp)

  This function is used to get the session interface used to indicate the interface which the BFD session is running on.

- s_int32_t smi_bfd_get_sess_addr_type_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_addr_type, bool_t snmp)

  This function is used to get the session address type of the BFD session.

- s_int32_t smi_bfd_get_sess_gtsm_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_gtsm, bool_t snmp)

  This function is used to get the session GTSM(Generalized TTL Security Mechanism) whether is enabled or not.

- s_int32_t smi_bfd_sess_gtsm_ttl_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_gtsm_ttl, bool_t snmp)

  This function is used to get the session GTSM(Generalized TTL Security Mechanism) TTL(Time to live) which specifies the minimum allowed TTL for received BFD control packets.GTSM TTL valid only when GTSM is enabled.

- s_int32_t smi_bfd_get_sess_dsrd_min_tx_intvl_sdkapi (struct smiclient_-globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_dsrd_min_tx_intvl, bool_t snmp)

  This function is used to get the session desired minimum transmission interval, which is the minimum interval,in microseconds, that the local system would like to use when transmitting BFD Control packets.

- s_int32_t smi_bfd_get_sess_req_min_rx_intvl_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_req_-min_rx_intvl, bool_t snmp)

  This function is used to get the session required minimum receive interval, which specifies the minimum interval, in microseconds, between received BFD Control packets the system is capable of supporting.

- s_int32_t smi_bfd_get_sess_req_min_echo_rx_intvl_sdkapi (struct smiclient_-globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_req_min_echo_rx_intvl, bool_t snmp)

  This function is used to get the session required minimum echo receive interval, which specifies the minimum interval, in microseconds, between received BFD Echo packets system is capable of supporting.

- s_int32_t smi_bfd_get_sess_detectmult_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_detectmult, bool_t snmp)

  This function is used to get the session detect time multiplier.

- s_int32_t smi_bfd_get_sess_neg_intvl_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_neg_intvl, bool_t snmp)

  This function is used to get the session negotiated interval in microseconds, that the local system is transmitting BFD Control packets.

- s_int32_t smi_bfd_get_sess_neg_echo_intvl_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_neg_-echo_intvl, bool_t snmp)

  This function is used to get the session negotiated echo interval in microseconds, that the local system is transmitting BFD echo packets.

- s_int32_t smi_bfd_get_sess_neg_detect_mult_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_neg_-detect_mult, bool_t snmp)

  This function is used to get the session negotiated detect multiplier.

- s_int32_t smi_bfd_get_sess_auth_pres_flag_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_-auth_pres_flag, bool_t snmp)

  This function is used to get the session authentication preserve flag which indicates that the local system's desire to use Authentication.

- s_int32_t smi_bfd_api_sess_auth_type_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_auth_type, bool_t snmp)

  This function is used to get the session authentication type.

- s_int32_t smi_bfd_api_sess_auth_key_id_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_auth_key_-id, bool_t snmp)

  This function is used to get the session authentication key ID. It permits multiple keys to be active simultaneously.

- s_int32_t smi_bfd_get_sess_stor_type_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_stor_type, bool_t snmp)

  This function is used to get the session storage type.

- s_int32_t smi_bfd_get_sess_row_status_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_row_status, bool_t snmp)

  This function is used to get the session row status.

- s_int32_t smi_bfd_get_perf_pkt_in_sdkapi (struct smiclient_globals ∗azg, u_-int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗perf_pkt_in, bool_t snmp)

  This function is used to get total number of BFD control messages received for the BFD session.

- s_int32_t smi_bfd_get_perf_pkt_out_sdkapi (struct smiclient_globals ∗azg, u_-int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗perf_pkt_out, bool_t snmp)

  This function is used to get total number of BFD control messages sent for the BFD session.

- s_int32_t smi_bfd_get_sess_up_time_sdkapi (struct smiclient_globals ∗azg, u_-int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗sess_up_time, bool_t snmp)

  This function is used to get value of sysUpTime on the most recent occasion at which the session came up.

- s_int32_t smi_bfd_get_perf_lastses_down_time_sdkapi (struct smiclient_-globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗perf_lastses_down_time, bool_t snmp)

  This function is used to get value of sysUpTime on the most recent occasion at which the last time communication was lost with the neighbor.

- s_int32_t smi_bfd_get_perf_lastcomm_lost_diag_sdkapi (struct smiclient_-globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗perf_lastcomm_lost_diag, bool_t snmp)

This function is used to get the BFD diag code for the last time communication was lost with the neighbor.

- s_int32_t smi_bfd_get_perf_sess_up_count_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗perf_-sess_up_count, bool_t snmp)

  This function is used to get number of times the session has gone into the Up state since the system last rebooted.

- s_int32_t smi_bfd_get_perf_disc_time_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗perf_disc_time, bool_t snmp)

  This function is used to get value of sysUpTime on the most recent occasion which any one or more of the session counters suffered a discontinuity.

- s_int32_t smi_bfd_get_perf_pkt_in_hc_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, ut_int64_t ∗perf_pkt_in_-hc, bool_t snmp)

  This function is used to get value represents the total number of BFD control messages received with respect to high capacity for the BFD session.

- s_int32_t smi_bfd_get_perf_pkt_out_hc_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, ut_int64_t ∗perf_pkt_out_-hc, bool_t snmp)

  This function is used to get value represents the total number of BFD control messages sent with respect to high capacity for the BFD session.

- s_int32_t smi_bfd_get_discmap_index_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t ∗discmap_index, bool_t snmp)

  This function is used to get the BfdSessIndexTC value referred to by the indices of the Discriminator Mapping entry.

- s_int32_t smi_bfd_set_sess_version_no_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t sess_version_no, bool_t snmp)

  This function is used to set the version number of the BFD protocol.

- s_int32_t smi_bfd_set_sess_src_udp_port_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t sess_src_-udp_port, bool_t snmp)

  This function is used to set the source UDP port number used for the BFD session's control packets.

- s_int32_t smi_bfd_set_sess_echo_src_udp_port_sdkapi (struct smiclient_-globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t sess_echo_src_udp_port, bool_t snmp)

  This function is used to set the source UDP port number used for BFD session's echo packets.

- s_int32_t smi_bfd_set_sess_admin_status_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t set_sess_-admin_status, bool_t snmp)

  This function is used to set the session administration status.

- s_int32_t smi_bfd_set_sess_dmnd_mode_dsrd_flag_sdkapi (struct smiclient_-globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t sess_dmnd_mode_dsrd_flag, bool_t snmp)

  This function is used to set the session demand mode desired flag which indicates that the local system's desire to use Demand mode.

- s_int32_t smi_bfd_set_sess_interface_sdkapi (struct smiclient_globals ∗azg, u_-int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t sess_interface, bool_t snmp)

  This function is used to get the session interface used to indicate the interface which the BFD session is running on.

- s_int32_t smi_bfd_set_sess_addr_type_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t sess_addr_type, bool_t snmp)

  This function is used to get the session address type of the BFD session.

- s_int32_t smi_bfd_set_sess_gtsm_sdkapi_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t sess_gtsm, bool_t snmp)

  This function is used to set the session GTSM(Generalized TTL Security Mechanism) enable/disable mode.

- s_int32_t smi_bfd_set_sess_gtsm_ttl_sdkapi (struct smiclient_globals ∗azg, u_-int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t sess_gtsm_ttl, bool_t snmp)

  This function is used to set the session GTSM(Generalized TTL Security Mechanism) TTL(Time to live) which specifies the minimum allowed TTL for received BFD control packets.

- s_int32_t smi_bfd_set_sess_dsrd_min_tx_intvl_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t sess_dsrd_-min_tx_intvl, bool_t snmp)

  This function is used to set the session desired minimum transmission interval.

- s_int32_t smi_bfd_set_sess_req_min_rx_intvl_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t sess_req_-min_rx_intvl, bool_t snmp)

  This function is used to set the session required minimum receive interval, which specifies the minimum interval in microseconds. between received BFD Echo packets system is capable of supporting.

- s_int32_t smi_bfd_set_sess_req_min_echo_rx_intvl_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t sess_req_min_echo_rx_intvl, bool_t snmp)

  This function is used to set the session required minimum echo receive interval, which specifies the minimum interval, in microseconds, between received BFD Echo packets system is capable of supporting.

- s_int32_t smi_bfd_set_sess_detect_mult_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t set_sess_detect_mult, bool_t snmp)

  This function is used to set the session detect time multiplier.

- s_int32_t smi_bfd_set_sess_stor_type_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t sess_stor_type, bool_t snmp)

  This function is used to set the session storage type.

- s_int32_t smi_bfd_set_sess_row_status_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t index, u_int32_t sess_row_status, bool_t snmp)

  This function is used to get the session row status.

- s_int32_t smi_bfd_process_set (struct smiclient_globals ∗azg, u_int32_t vr_id, int proc_id)

  Creates a new Bidirectional Forward Detection (BFD) process instance, if not before created, else returns the existing one.

- s_int32_t smi_bfd_proto_interval_set_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, char ∗ifname, u_int32_t min_tx, u_int32_t min_rx, u_int32_t multiplier)

  Sets different time intervals of singlehop BFD in microseconds.

- s_int32_t smi_bfd_proto_interval_unset_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, char ∗ifname)

  Unsets different time intervals of BFD.

- s_int32_t smi_bfd_multihop_proto_interval_set_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, char ∗ifname, struct pal_in4_addr ip_addr, u_int32_t min_tx, u_int32_t min_rx, u_int32_t multiplier)

  Sets different time intervals of multihop BFD in microseconds.

- s_int32_t smi_bfd_multihop_proto_interval_unset_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, char ∗ifname, struct pal_in4_addr ipv4)

  Unsets different time intervals of multihop BFD in microseconds.

- s_int32_t smi_bfd_echo_mode_set (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id)

Sets this BFD session to run in Echo mode at switch level.

- s_int32_t smi_bfd_echo_mode_unset (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id)

  Unsets this BFD session to run in Echo mode at switch level.

- s_int32_t smi_bfd_proto_slow_timer_set (struct smiclient_globals ∗azg, u_-int32_t vr_id, s_int32_t proc_id, u_int32_t slow_interval)

  Sets this BFD session's slow time interval in microseconds at switch level.

- s_int32_t smi_bfd_proto_slow_timer_unset (struct smiclient_globals ∗azg, u_-int32_t vr_id, s_int32_t proc_id)

  Unsets this BFD session's slow time interval in microseconds at switch level.

- s_int32_t smi_bfd_interface_echo_mode_set_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, char ∗ifname)

  Sets this BFD session to run in Echo mode at interface level.

- s_int32_t smi_bfd_interface_echo_mode_unset_sdkapi (struct smiclient_-globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, char ∗ifname)

  Unsets this BFD session to run in Echo mode at interface level.

- s_int32_t smi_bfd_interface_slow_timer_set_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, char ∗ifname, u_int32_t slow_-interval)

  Sets this BFD session's slow time interval in microseconds at interface level.

- s_int32_t smi_bfd_interface_slow_timer_unset_sdkapi (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, char ∗ifname)

  Unsets this BFD session's slow time interval in microseconds at interface level.

- s_int32_t smi_bfd_add_user_session (struct smiclient_globals ∗azg, u_int32_-t vr_id, s_int32_t proc_id, struct pal_in4_addr ∗src_addr, struct pal_in4_addr ∗dst_addr, s_int32_t ifindex, u_int32_t flags)

  This function adds an IPv4 BFD user session.

- s_int32_t smi_bfd_del_user_session (struct smiclient_globals ∗azg, u_int32_-t vr_id, s_int32_t proc_id, struct pal_in4_addr ∗src_addr, struct pal_in4_addr ∗dst_addr, s_int32_t ifindex, u_int32_t flags)

  This function deletes an IPv4 BFD user session.

- s_int32_t smi_bfd_echo_interval_set (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, char ∗name, u_int32_t echo_tx)

  This function sets the BFD echo mode transmission interval for all single-hop sessions on an interface.

- s_int32_t smi_bfd_echo_interval_unset (struct smiclient_globals ∗azg, u_-int32_t vr_id, s_int32_t proc_id, char ∗name)

This function resets the BFD echo mode transmission interval to its default value for all sessions on an interface.

- s_int32_t smi_bfd_proto_auth_set (struct smiclient_globals ∗azg, u_int32_t vr_-id, s_int32_t proc_id, char ∗ifname, char ∗auth_type_str, u_int32_t key_id, char ∗key_str, char ∗key_chain)

  This function adds an IPv6 BFD user session.

- s_int32_t smi_bfd_proto_auth_unset (struct smiclient_globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t ifindex)

  This function unsets the authentication information on an interface having sessions.

- s_int32_t smi_bfd_proto_multihop_auth_ipv4_set_sdkapi (struct smiclient_-globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, u_int32_t ifindex, struct pal_-in4_addr ∗ipv4_addr, char ∗auth_type_str, u_int32_t key_id, char ∗key_str, char ∗key_chain)

  This function sets the authentication information for a multihop session.

- s_int32_t smi_bfd_proto_multihop_auth_unset_ipv4_sdkapi (struct smiclient_-globals ∗azg, u_int32_t vr_id, s_int32_t proc_id, s_int32_t ifindex, struct pal_-in4_addr ∗ipv4_addr)

  This function unsets the authentication information from a multihop session.

2.1.1 Detailed Description

Provides APIs for managing Bidirectional Forwarding Detection(BFD) in ZebOS. The ZebOS Bidirectional Forwarding Detection (BFD) module is designed to work with most router architectures wherever hardware supports some level of Bidirectional Forwarding Detection capabilities, and in situations where there is no hardware support at all. The BFD module is designed to work in conjunction with application protocol modules (for example, OSPF, BGP, RIP) to enable them to configure BFD sessions and for the sessions to get the bidirectional forwarding failure notifications from BFD. The way each application reacts to a session-down event is application-specific.

2.1.2 Function Documentation

2.1.2.1 s_int32_t smi_bfd_add_user_session (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, struct pal_in4_addr ∗ *src_addr*, struct pal_in4_addr ∗ *dst_addr*, s_int32_t *ifindex*, u_int32_t *flags*)

This function adds an IPv4 BFD user session. smi_bfd_add_user_session

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***src_addr*** Source address for the session

← ***dst_addr*** Destination address for the session

← ***ifindex*** Interface index

← ***flags*** Define the properties of a session
BFD_MSG_SESSION_FLAG_MH(Multi-Hop)
BFD_MSG_SESSION_FLAG_DC(Demand Circuit)
BFD_MSG_SESSION_FLAG_PS(Persistent Session)
BFD_MSG_SESSION_FLAG_AD(User Admin Down)

Returns:

BFD_SUCCESS on success, otherwise one of the following errors BFD_API_-SET_ERR_VR_NOT_EXIST
BFD_API_INSTANCE_NOT_FOUND
BFD_API_INVALID_ADDR
BFD_FAILURE

2.1.2.2 s_int32_t smi_bfd_api_get_sess_diag_sdkapi (struct smiclient_globals * *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t * *sess_diag*, bool_t *snmp*)

This function is used to get the session diagram, a diagnostic code specifying the local system's reason for the last transition of the session from up to some other state. smi_-bfd_api_get_sess_diag_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***sess_diag*** The session diagram (diagnostic code) numeric <0-10>
0 - bfd_diag_no_diag, 1 - bfd_diag_ctrl_detect_exp,
2 - bfd_diag_echo_failed, 3- bfd_diag_nbr_session_down,
4 - bfd_diag_fwd_reset, 5 - bfd_diag_path_down,
6 - bfd_diag_concat_path_down, 7 - bfd_diag_admin_down,
8 - bfd_diag_rev_concat_path_down, 9 - bfd_diag_config_error,
10 - bfd_diag_not_forwarding

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.3 s_int32_t smi_bfd_api_sess_auth_key_id_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *sess_auth_key_id*, bool_t *snmp*)

This function is used to get the session authentication key ID. It permits multiple keys to be active simultaneously. smi_bfd_api_sess_auth_key_id_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***sess_auth_key_id*** Session authentication type
-1 - BFD_AUTH_KEY_ID

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.4 s_int32_t smi_bfd_api_sess_auth_type_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *sess_auth_type*, bool_t *snmp*)

This function is used to get the session authentication type. smi_bfd_api_sess_auth_-type_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***sess_auth_type*** Session authentication type
0 - BFD_AUTH_TYPE_RESERVED

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.5 s_int32_t smi_bfd_del_user_session (struct smiclient_globals * *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, struct pal_in4_addr * *src_addr*, struct pal_in4_addr * *dst_addr*, s_int32_t *ifindex*, u_int32_t *flags*)

This function deletes an IPv4 BFD user session. smi_bfd_del_user_session

Parameters:

← ***azg*** Pointer to the SMI client global structure
← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation
← ***proc_id*** Process ID for the BFD instance numeric <0-65535>
← ***src_addr*** Source address for the session
← ***dst_addr*** Destination address for the session
← ***ifindex*** Interface index
← ***flags*** Define the properties of a session
BFD_MSG_SESSION_FLAG_MH(Multi-Hop)
BFD_MSG_SESSION_FLAG_DC(Demand Circuit)
BFD_MSG_SESSION_FLAG_PS(Persistent Session)
BFD_MSG_SESSION_FLAG_AD(User Admin Down)

Returns:

BFD_SUCCESS on success, otherwise one of the following errors BFD_API_-SET_ERR_VR_NOT_EXIST
BFD_API_INSTANCE_NOT_FOUND
BFD_API_INVALID_ADDR
BFD_FAILURE

2.1.2.6 s_int32_t smi_bfd_echo_interval_set (struct smiclient_globals * *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, char * *name*, u_int32_t *echo_tx*)

This function sets the BFD echo mode transmission interval for all single-hop sessions on an interface. smi_bfd_echo_interval_set

Parameters:

← ***azg*** Pointer to the SMI client global structure
← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation
← ***proc_id*** Process ID for the BFD instance numeric <0-65535>
← ***name*** Interface name
← ***echo_tx*** Echo transmission interval

Returns:

BFD_SUCCESS on success, otherwise one of the following errors BFD_API_-SET_ERR_VR_NOT_EXIST
BFD_API_INSTANCE_NOT_FOUND
BFD_API_INVALID_ADDR
BFD_FAILURE

2.1.2.7 s_int32_t smi_bfd_echo_interval_unset (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, char ∗ *name*)

This function resets the BFD echo mode transmission interval to its default value for all sessions on an interface. smi_bfd_echo_interval_unset

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***name*** Interface name

Returns:

BFD_SUCCESS on success, otherwise one of the following errors BFD_API_-SET_ERR_VR_NOT_EXIST
BFD_API_INSTANCE_NOT_FOUND
BFD_API_INVALID_ADDR
BFD_FAILURE

2.1.2.8 s_int32_t smi_bfd_echo_mode_set (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*)

Sets this BFD session to run in Echo mode at switch level. smi_bfd_echo_mode_set

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

Returns:

0 on success, otherwise one of the following error codes
BFD_API_INSTANCE_NOT_FOUND
BFD_API_SET_ERROR

2.1.2.9 s_int32_t smi_bfd_echo_mode_unset (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*)

Unsets this BFD session to run in Echo mode at switch level. smi_bfd_echo_mode_-unset

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

Returns:

0 on success, otherwise one of the following error codes
BFD_API_INSTANCE_NOT_FOUND
BFD_API_SET_ERROR

2.1.2.10 s_int32_t smi_bfd_get_discmap_index_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *discmap_index*, bool_t *snmp*)

This function is used to get the BfdSessIndexTC value referred to by the indices of the Discriminator Mapping entry. smi_bfd_get_discmap_index_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***discmap_index*** BfdSessIndexTC value

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.11 s_int32_t smi_bfd_get_perf_disc_time_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *perf_disc_time*, bool_t *snmp*)

This function is used to get value of sysUpTime on the most recent occasion which any one or more of the session counters suffered a discontinuity. smi_bfd_get_perf_disc_-time_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***perf_disc_time*** Value of sysUpTime when one or more of the session counters suffered a discontinuity

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.12 s_int32_t smi_bfd_get_perf_lastcomm_lost_diag_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *perf_lastcomm_lost_diag*, bool_t *snmp*)

This function is used to get the BFD diag code for the last time communication was lost with the neighbor. smi_bfd_get_perf_lastcomm_lost_diag_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***perf_lastcomm_lost_diag*** The session diagram (diagnostic code) numeric <0-10>
0 - bfd_diag_no_diag, 1 - bfd_diag_ctrl_detect_exp,
2 - bfd_diag_echo_failed, 3- bfd_diag_nbr_session_down,
4 - bfd_diag_fwd_reset, 5 - bfd_diag_path_down,
6 - bfd_diag_concat_path_down, 7 - bfd_diag_admin_down,
8 - bfd_diag_rev_concat_path_down, 9 - bfd_diag_config_error,
10 - bfd_diag_not_forwarding

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.13 s_int32_t smi_bfd_get_perf_lastses_down_time_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *perf_lastses_down_time*, bool_t *snmp*)

This function is used to get value of sysUpTime on the most recent occasion at which the last time communication was lost with the neighbor. smi_bfd_get_perf_lastses_-down_time_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***perf_lastses_down_time*** Value of sysUpTime at which the last time communication was lost with neighbor

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.14 s_int32_t smi_bfd_get_perf_pkt_in_hc_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, ut_int64_t ∗ *perf_pkt_in_hc*, bool_t *snmp*)

This function is used to get value represents the total number of BFD control messages received with respect to high capacity for the BFD session. smi_bfd_get_perf_pkt_-in_hc_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***perf_pkt_in_hc*** Total number of BFD control messages received

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.15 s_int32_t smi_bfd_get_perf_pkt_in_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *perf_pkt_in*, bool_t *snmp*)

This function is used to get total number of BFD control messages received for the BFD session. smi_bfd_get_perf_pkt_in_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***perf_pkt_in*** Number of packets received

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.16 s_int32_t smi_bfd_get_perf_pkt_out_hc_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, ut_int64_t ∗ *perf_pkt_out_hc*, bool_t *snmp*)

This function is used to get value represents the total number of BFD control messages sent with respect to high capacity for the BFD session. smi_bfd_get_perf_pkt_out_-hc_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***perf_pkt_out_hc*** Total number of BFD control messages sent

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.17 s_int32_t smi_bfd_get_perf_pkt_out_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *perf_pkt_out*, bool_t *snmp*)

This function is used to get total number of BFD control messages sent for the BFD session. smi_bfd_get_perf_pkt_out_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***perf_pkt_out*** Number of packets sent

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.18 s_int32_t smi_bfd_get_perf_sess_up_count_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *perf_sess_up_count*, bool_t *snmp*)

This function is used to get number of times the session has gone into the Up state since the system last rebooted. smi_bfd_get_perf_sess_up_count_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

→ ***perf_sess_up_count*** Number of times the session has gone into the Up state

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.19 s_int32_t smi_bfd_get_sess_addr_type_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *sess_addr_type*, bool_t *snmp*)

This function is used to get the session address type of the BFD session. smi_bfd_-get_sess_addr_type_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***sess_addr_type*** Session address type numeric <0-4>

0 - ADDR_TYPE_unknown

1 - ADDR_TYPE_ipv4

2 - ADDR_TYPE_ipv6

3 - ADDR_TYPE_ipv4z

4 - ADDR_TYPE_ipv6z

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)

0 - PAL_FALSE (SNMP disabled)

1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.20 s_int32_t smi_bfd_get_sess_admin_status_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *sess_admin_status*, bool_t *snmp*)

This function is used to get the session administration status.A transition from 'stop' to 'start' will start the BFD state machine for the session. The state machine will have an initial state of down. smi_bfd_get_sess_admin_status_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***sess_admin_status*** Session administration status

← ***snmp*** SNMP Enable/Disable numeric <0 | 1>

0 - PAL_FALSE (SNMP disabled)

1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.21 s_int32_t smi_bfd_get_sess_auth_pres_flag_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *sess_auth_pres_flag*, bool_t *snmp*)

This function is used to get the session authentication preserve flag which indicates that the local system's desire to use Authentication. smi_bfd_get_sess_auth_pres_-flag_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***sess_auth_pres_flag*** Session authentication preserve flag value numeric (1 | 2)
1 - BFD_API_TRUE (If BFD session must be authenticated)
2 - BFD_API_FALSE (If BFD session doesnot need to be authenticated)

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.22 s_int32_t smi_bfd_get_sess_cntrlplane_indep_flag_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *sess_cntrlplane_indep_flag*, bool_t *snmp*)

This function is used to get the session control plane independent flag indicates that the local system's ability to continue to function through a disruption of the control plane. smi_bfd_get_sess_cntrlplane_indep_flag_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***sess_cntrlplane_indep_flag*** Control plane independent flag of the session numeric (0 | 1)
0 - PAL_FALSE
1 - PAL_TRUE

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.23 s_int32_t smi_bfd_get_sess_dest_udp_port_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *sess_dest_udp_port*, bool_t *snmp*)

This function is used to get the destination UDP port number used for the BFD session's control packets. smi_bfd_get_sess_dest_udp_port_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***sess_dest_udp_port*** Destination UDP port number of the session

← ***snmp*** SNMP Enable/Disable numeric <0 | 1>
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.24 s_int32_t smi_bfd_get_sess_detectmult_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *sess_detectmult*, bool_t *snmp*)

This function is used to get the session detect time multiplier. smi_bfd_get_sess_-detectmult_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***sess_detectmult*** The detect multiple value of the session

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.25 s_int32_t smi_bfd_get_sess_disc_sdkapi (struct smiclient_globals * *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t * *sess_disc*, bool_t *snmp*)

This function is used to get the session discriminator for an SNMP Get request. smi_-bfd_get_sess_disc_sdkapi

Parameters:

$\leftarrow$ ***azg*** Pointer to the SMI client global structure

$\leftarrow$ ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

$\leftarrow$ ***proc_id*** Process ID for the BFD instance numeric <0-65535>

$\leftarrow$ ***index*** BFD session index value

$\rightarrow$ ***sess_disc*** Local session Discriminator

$\leftarrow$ ***snmp*** SNMP Enable/Disable numeric <0 | 1>
0 - PAL_FALSE (SNMP disabled) 1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.26 s_int32_t smi_bfd_get_sess_dmnd_mode_dsrd_flag_sdkapi (struct smiclient_globals * *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t * *sess_dmnd_mode_dsrd_flag*, bool_t *snmp*)

This function is used to get the session demand mode desired flag which indicates that the local system's desire to use Demand mode. smi_bfd_get_sess_dmnd_mode_dsrd_-flag_sdkapi

Parameters:

$\leftarrow$ ***azg*** Pointer to the SMI client global structure

$\leftarrow$ ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

$\leftarrow$ ***proc_id*** Process ID for the BFD instance numeric <0-65535>

$\leftarrow$ ***index*** BFD session index value

$\rightarrow$ ***sess_dmnd_mode_dsrd_flag*** Demand mode desired flag of the session (1 | 2)
1 - BFD_API_TRUE
2 - BFD_API_FALSE

$\leftarrow$ ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.27 s_int32_t smi_bfd_get_sess_dsrd_min_tx_intvl_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *sess_dsrd_min_tx_intvl*, bool_t *snmp*)

This function is used to get the session desired minimum transmission interval, which is the minimum interval,in microseconds, that the local system would like to use when transmitting BFD Control packets. smi_bfd_get_sess_dsrd_min_tx_intvl_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***sess_dsrd_min_tx_intvl*** Minimum transmission interval value of the session

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.28 s_int32_t smi_bfd_get_sess_echo_src_udp_port_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *sess_echo_src_udp_port*, bool_t *snmp*)

This function is used to get the source UDP port number used for BFD session's echo packets. smi_bfd_get_sess_echo_src_udp_port_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***sess_echo_src_udp_port*** Echo source UDP port number of the session

← ***snmp*** SNMP Enable/Disable numeric <0 | 1>
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.29 s_int32_t smi_bfd_get_sess_gtsm_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *sess_gtsm*, bool_t *snmp*)

This function is used to get the session GTSM(Generalized TTL Security Mechanism) whether is enabled or not. smi_bfd_get_sess_gtsm_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***sess_gtsm*** GTSM status numeric (0 | 1)
0 - PAL_FALSE (GTSM disabled) 1 - PAL_TRUE (GTSM enabled)

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.30 s_int32_t smi_bfd_get_sess_interface_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *sess_interface*, bool_t *snmp*)

This function is used to get the session interface used to indicate the interface which the BFD session is running on. smi_bfd_get_sess_interface_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***sess_interface*** Session interface

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.31 s_int32_t smi_bfd_get_sess_mh_unlnk_mode_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *mh_unlnk_mode*, bool_t *snmp*)

This function is used to get the session multihop UNI(Unidirectional) link mode for SNMP Get request. smi_bfd_get_sess_mh_unlnk_mode_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***mh_unlnk_mode*** Multihop UNI link mode of the session numeric <1-3>
1 - BFD_MH_UNLINK_MODE_NONE
2 - BFD_MH_UNLINK_MODE_ACTIVE
3 - BFD_MH_UNLINK_MODE_PASSIVE

← ***snmp*** SNMP Enable/Disable numeric <0 | 1>
0 - PAL_FALSE (SNMP disabled) 1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.32 s_int32_t smi_bfd_get_sess_neg_detect_mult_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *sess_neg_detect_mult*, bool_t *snmp*)

This function is used to get the session negotiated detect multiplier. smi_bfd_get_-sess_neg_detect_mult_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***sess_neg_detect_mult*** Negotiated detect multiple value

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.33 s_int32_t smi_bfd_get_sess_neg_echo_intvl_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *sess_neg_echo_intvl*, bool_t *snmp*)

This function is used to get the session negotiated echo interval in microseconds, that the local system is transmitting BFD echo packets. smi_bfd_get_sess_neg_echo_-intvl_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***sess_neg_echo_intvl*** Negotiated echo interval of the session

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.34 s_int32_t smi_bfd_get_sess_neg_intvl_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *sess_neg_intvl*, bool_t *snmp*)

This function is used to get the session negotiated interval in microseconds, that the local system is transmitting BFD Control packets. smi_bfd_get_sess_neg_intvl_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***sess_neg_intvl*** Negotiated interval of the session

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.35 s_int32_t smi_bfd_get_sess_oper_mode_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *sess_oper_mode*, bool_t *snmp*)

This function is used to get the current operating mode that BFD session is operating in. smi_bfd_get_sess_oper_mode_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure
← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation
← ***proc_id*** Process ID for the BFD instance numeric <0-65535>
← ***index*** BFD session index value
→ ***sess_oper_mode*** Operational mode of the session numeric <1-4>
1 - BFD_ASYN_ECHO_MODE
2 - BFD_ASYN_WO_ECHO_MODE
3 - BFD_DMND_ECHO_MODE
4 - BFD_DMND_WO_ECHO_MODE
← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.36 s_int32_t smi_bfd_get_sess_req_min_echo_rx_intvl_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *sess_req_min_echo_rx_intvl*, bool_t *snmp*)

This function is used to get the session required minimum echo receive interval, which specifies the minimum interval, in microseconds, between received BFD Echo packets system is capable of supporting. smi_bfd_get_sess_req_min_echo_rx_intvl_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure
← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation
← ***proc_id*** Process ID for the BFD instance numeric <0-65535>
← ***index*** BFD session index value
→ ***sess_req_min_echo_rx_intvl*** Required minimum echo receive interval of the session
← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.37 s_int32_t smi_bfd_get_sess_req_min_rx_intvl_sdkapi (struct smiclient_globals $*$ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t $*$ *sess_req_min_rx_intvl*, bool_t *snmp*)

This function is used to get the session required minimum receive interval, which specifies the minimum interval, in microseconds, between received BFD Control packets the system is capable of supporting. smi_bfd_get_sess_req_min_rx_intvl_sdkapi

Parameters:

$\leftarrow$ ***azg*** Pointer to the SMI client global structure

$\leftarrow$ ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

$\leftarrow$ ***proc_id*** Process ID for the BFD instance numeric <0-65535>

$\leftarrow$ ***index*** BFD session index value

$\rightarrow$ ***sess_req_min_rx_intvl*** Required minimum receive interval of the session

$\leftarrow$ ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.38 s_int32_t smi_bfd_get_sess_rmte_disc_sdkapi (struct smiclient_globals $*$ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t $*$ *sess_rmte_disc*, bool_t *snmp*)

This function is used to get the session discriminator chosen by the remote system for the BFD session. smi_bfd_get_sess_rmte_disc_sdkapi

Parameters:

$\leftarrow$ ***azg*** Pointer to the SMI client global structure

$\leftarrow$ ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

$\leftarrow$ ***proc_id*** Process ID for the BFD instance numeric <0-65535>

$\leftarrow$ ***index*** BFD session index value

$\rightarrow$ ***sess_rmte_disc*** Session remote discriminator

$\leftarrow$ ***snmp*** SNMP Enable/Disable numeric <0 | 1>
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.39 s_int32_t smi_bfd_get_sess_rmte_heard_flag_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *sess_rmte_heard_flag*, bool_t *snmp*)

This function is used to get the status of BFD packet reception from the remote system. Specifically, it is set to true if the local system is actively receiving BFD packets from the remote system, and is set to false(2) if the local system has not received BFD packets recently. smi_bfd_get_sess_rmte_heard_flag_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***sess_rmte_heard_flag*** Remote heard flag of the session numeric (1 | 2)
1 - BFD_API_TRUE 2 - BFD_API_FALSE

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.40 s_int32_t smi_bfd_get_sess_row_status_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *sess_row_status*, bool_t *snmp*)

This function is used to get the session row status. smi_bfd_get_sess_row_status_-sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***sess_row_status*** Session row status

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.41 s_int32_t smi_bfd_get_sess_src_udp_port_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *sess_src_udp_port*, bool_t *snmp*)

This function is used to get the source UDP port number used for the BFD session's control packets. smi_bfd_get_sess_src_udp_port_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***sess_src_udp_port*** Source UDP port number of the session

← ***snmp*** SNMP Enable/Disable numeric <0 | 1>
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.42 s_int32_t smi_bfd_get_sess_state_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *sess_state*, bool_t *snmp*)

This function is used to get the session state. smi_bfd_get_sess_state_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***sess_state*** Session status numeric<1-4>
1 - BFD_API_SESS_ST_AD_DWN 2 - BFD_API_SESS_ST_DWN 3 - BFD_API_SESS_ST_INIT 4 - BFD_API_SESS_ST_UP

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.43 s_int32_t smi_bfd_get_sess_stor_type_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *sess_stor_type*, bool_t *snmp*)

This function is used to get the session storage type. smi_bfd_get_sess_stor_type_-sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***sess_stor_type*** Session storage type

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.44 s_int32_t smi_bfd_get_sess_type_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *session_type*, bool_t *snmp*)

This function is used to get the session type for an SNMP Get request. smi_bfd_get_-sess_type_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***session_type*** Type of BFD session type

← ***snmp*** SNMP Enable/Disable numeric <0 | 1>
0 - PAL_FALSE (SNMP disabled) 1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.45 s_int32_t smi_bfd_get_sess_up_time_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *sess_up_time*, bool_t *snmp*)

This function is used to get value of sysUpTime on the most recent occasion at which the session came up. smi_bfd_get_sess_up_time_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***sess_up_time*** Value of sysUpTime at which session came up

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.46 s_int32_t smi_bfd_get_sess_version_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t ∗ *version*, bool_t *snmp*)

This function gets the version number of the BFD protocol. smi_bfd_get_sess_-version_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***version*** Version number

← ***snmp*** SNMP Enable/Disable numeric <0 | 1>
0 - PAL_FALSE (SNMP disabled) 1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.47 s_int32_t smi_bfd_interface_echo_mode_set_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, char ∗ *ifname*)

Sets this BFD session to run in Echo mode at interface level. smi_bfd_interface_echo_-mode_set_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***ifname*** Interface name

Returns:

0 on success, otherwise one of the following error codes
BFD_API_SET_ERR_VR_NOT_EXIST
BFD_API_INSTANCE_NOT_FOUND
BFD_API_IF_NOT_FOUND
BFD_API_SET_ERROR

2.1.2.48 s_int32_t smi_bfd_interface_echo_mode_unset_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, char ∗ *ifname*)

Unsets this BFD session to run in Echo mode at interface level. smi_bfd_interface_-echo_mode_unset_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***ifname*** Interface name

Returns:

0 on success, otherwise one of the following error codes
BFD_API_SET_ERR_VR_NOT_EXIST
BFD_API_INSTANCE_NOT_FOUND
BFD_API_IF_NOT_FOUND
BFD_API_SET_ERROR

2.1.2.49 s_int32_t smi_bfd_interface_slow_timer_set_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, char ∗ *ifname*, u_int32_t *slow_interval*)

Sets this BFD session's slow time interval in microseconds at interface level. smi_-bfd_interface_slow_timer_set_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***ifname*** Interface name

← ***slow_interval*** Slow timer inreval <1000-4294967000>

Returns:

0 on success, otherwise one of the following error codes
BFD_API_SET_ERR_VR_NOT_EXIST
BFD_API_INSTANCE_NOT_FOUND
BFD_API_IF_NOT_FOUND
BFD_API_SET_ERROR

2.1.2.50 s_int32_t smi_bfd_interface_slow_timer_unset_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, char ∗ *ifname*)

Unsets this BFD session's slow time interval in microseconds at interface level. smi_-bfd_interface_slow_timer_unset_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***ifname*** Interface name

Returns:

0 on success, otherwise one of the following error codes
BFD_API_SET_ERR_VR_NOT_EXIST
BFD_API_INSTANCE_NOT_FOUND
BFD_API_IF_NOT_FOUND
BFD_API_SET_ERROR

2.1.2.51 s_int32_t smi_bfd_multihop_proto_interval_set_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, char ∗ *ifname*, struct pal_in4_addr *ip_addr*, u_int32_t *min_tx*, u_int32_t *min_rx*, u_int32_t *multiplier*)

Sets different time intervals of multihop BFD in microseconds. smi_bfd_multihop_-proto_interval_set_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***ifname*** Interface name

← ***ip_addr*** IPv4 Peer address

← ***min_tx*** Minimum BFD Control packets transmitting interval in microseconds <1000-4294967000>

← ***min_rx*** Minimum BFD Control packets receiving interval in microseconds <1000-4294967000>

← ***multiplier*** BFD failure detection multiplier <1-255>

Returns:

0 on success, otherwise one of the following error codes
BFD_API_SET_ERR_VR_NOT_EXIST
BFD_API_INSTANCE_NOT_FOUND
BFD_API_IF_NOT_FOUND
BFD_API_MH_NOT_FOUND
BFD_API_SET_ERROR

2.1.2.52 s_int32_t smi_bfd_multihop_proto_interval_unset_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, char ∗ *ifname*, struct pal_in4_addr *ipv4*)

Unsets different time intervals of multihop BFD in microseconds. smi_bfd_multihop_-proto_interval_unset_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***ifname*** Interface name

← ***ip_addr*** IPv4 Peer address

Returns:

0 on success, otherwise one of the following error codes
BFD_API_SET_ERR_VR_NOT_EXIST
BFD_API_INSTANCE_NOT_FOUND
BFD_API_IF_NOT_FOUND
BFD_API_MH_NOT_FOUND
BFD_API_SET_ERROR

2.1.2.53 s_int32_t smi_bfd_notification_set (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, u_int32_t *notif_flag*)

Sets to enable or disable the notification emission of BFD session on this device. If this object is set to true(1), then it enables the emission of bfdSessUp and bfdSessDown notifications; otherwise these notifications are not emitted. smi_bfd_notification_set

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***notif_flag*** Enable/Disable flag

Returns:

0 on success, otherwise one of the following error codes

2.1.2.54 s_int32_t smi_bfd_process_set (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, int *proc_id*)

Creates a new Bidirectional Forward Detection (BFD) process instance, if not before created, else returns the existing one. smi_bfd_process_set

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

Returns:

0 on success, otherwise one of the following error codes
BFD_API_SET_ERR_VR_NOT_EXIST
BFD_API_SET_ERR_PROCESS_ID_INVALID

2.1.2.55 s_int32_t smi_bfd_proto_auth_set (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, char ∗ *ifname*, char ∗ *auth_type_str*, u_int32_t *key_id*, char ∗ *key_str*, char ∗ *key_chain*)

This function adds an IPv6 BFD user session. smi_bfd_add_ipv6_user_session

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***ipv6_src_addr*** Source address for the session

← ***ipv6_dst_addr*** Destination address for the session

← ***ifindex*** Interface index

← ***flags*** Define the properties of a session
BFD_MSG_SESSION_FLAG_MH(Multi-Hop)
BFD_MSG_SESSION_FLAG_DC(Demand Circuit)
BFD_MSG_SESSION_FLAG_PS(Persistent Session)
BFD_MSG_SESSION_FLAG_AD(User Admin Down)

Returns:

BFD_SUCCESS on success, otherwise one of the following errors BFD_API_-SET_ERR_VR_NOT_EXIST
BFD_API_INSTANCE_NOT_FOUND
BFD_API_INVALID_ADDR
BFD_FAILURE

smi_bfd_proto_auth_set

This function sets the authentication information on an interface having sessions.

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***ifname*** Interface name string

← ***auth_type_str*** Authentication string type

← ***key_id*** Key identifier

← ***key_str*** Key string

← ***key_chain*** Key chain

Returns:

BFD_SUCCESS on success, otherwise one of the following errors BFD_API_-INVALID_AUTH_TYPE

BFD_API_INVALID_KEY_ID
BFD_API_SET_ERR_VR_NOT_EXIST
BFD_API_INSTANCE_NOT_FOUND
BFD_API_IF_NOT_FOUND

2.1.2.56 s_int32_t smi_bfd_proto_auth_unset (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *ifindex*)

This function unsets the authentication information on an interface having sessions. smi_bfd_proto_auth_unset

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***ifindex*** Interface index

Returns:

BFD_SUCCESS on success, otherwise one of the following errors BFD_API_-INVALID_AUTH_TYPE
BFD_API_INVALID_KEY_ID
BFD_API_SET_ERR_VR_NOT_EXIST
BFD_API_INSTANCE_NOT_FOUND
BFD_API_IF_NOT_FOUND

2.1.2.57 s_int32_t smi_bfd_proto_interval_set_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, char ∗ *ifname*, u_int32_t *min_tx*, u_int32_t *min_rx*, u_int32_t *multiplier*)

Sets different time intervals of singlehop BFD in microseconds. smi_bfd_proto_-interval_set_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***ifname*** Interface name

← ***min_tx*** Minimum BFD Control packets transmitting interval in microseconds <1000-4294967000>

← ***min_rx*** Minimum BFD Control packets receiving interval in microseconds <1000-4294967000>

← ***multiplier*** BFD failure detection multiplier <1-255>

Returns:

0 on success, otherwise one of the following error codes
BFD_API_SET_ERR_VR_NOT_EXIST
BFD_API_INSTANCE_NOT_FOUND
BFD_API_IF_NOT_FOUND
BFD_API_SET_ERROR

2.1.2.58 s_int32_t smi_bfd_proto_interval_unset_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, char ∗ *ifname*)

Unsets different time intervals of BFD. smi_bfd_proto_interval_unset_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***ifname*** Interface name

Returns:

0 on success, otherwise one of the following error codes
BFD_API_SET_ERR_VR_NOT_EXIST
BFD_API_IF_NOT_FOUND
BFD_API_SET_ERROR

2.1.2.59 s_int32_t smi_bfd_proto_multihop_auth_ipv4_set_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *ifindex*, struct pal_in4_addr ∗ *ipv4_addr*, char ∗ *auth_type_str*, u_int32_t *key_id*, char ∗ *key_str*, char ∗ *key_chain*)

This function sets the authentication information for a multihop session. smi_bfd_-proto_multihop_auth_ipv6_set_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***ifindex*** Interface index

← ***ipv6_addr*** ipv6 Address

← ***auth_type_str*** Authentication string type

← ***key_id*** Key identifier

← ***key_str*** Key string

← ***key_chain*** Key chain

Returns:

BFD_SUCCESS on success, otherwise one of the following errors BFD_API_-INVALID_AUTH_TYPE
BFD_API_MH_NOT_FOUND
BFD_API_SET_ERR_VR_NOT_EXIST
BFD_API_INSTANCE_NOT_FOUND
BFD_API_IF_NOT_FOUND

smi_bfd_proto_multihop_auth_ipv4_set_sdkapi

This function sets the authentication information for a multihop session.

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***ifindex*** Interface index

← ***ipv4_addr*** ipv4 Address

← ***auth_type_str*** Authentication string type

← ***key_id*** Key identifier

← ***key_str*** Key string

← ***key_chain*** Key chain

Returns:

BFD_SUCCESS on success, otherwise one of the following errors BFD_API_-INVALID_AUTH_TYPE
BFD_API_MH_NOT_FOUND
BFD_API_SET_ERR_VR_NOT_EXIST
BFD_API_INSTANCE_NOT_FOUND
BFD_API_IF_NOT_FOUND

2.1.2.60 s_int32_t smi_bfd_proto_multihop_auth_unset_ipv4_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, s_int32_t *ifindex*, struct pal_in4_addr ∗ *ipv4_addr*)

This function unsets the authentication information from a multihop session. smi_-bfd_proto_multihop_auth_unset_ipv4_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***ifindex*** Interface index

← ***ipv4_addr*** ipv4 Address

Returns:

BFD_SUCCESS on success, otherwise one of the following errors BFD_API_-INVALID_AUTH_TYPE
BFD_API_MH_NOT_FOUND
BFD_API_SET_ERR_VR_NOT_EXIST
BFD_API_INSTANCE_NOT_FOUND
BFD_API_IF_NOT_FOUND

2.1.2.61 s_int32_t smi_bfd_proto_slow_timer_set (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *slow_interval*)

Sets this BFD session's slow time interval in microseconds at switch level. smi_bfd_-proto_slow_timer_set

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***slow_interval*** Slow timer inreval <1000-4294967000>

Returns:

0 on success, otherwise one of the following error codes
BFD_API_INSTANCE_NOT_FOUND
BFD_API_SET_ERROR

2.1.2.62 s_int32_t smi_bfd_proto_slow_timer_unset (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*)

Unsets this BFD session's slow time interval in microseconds at switch level. smi_-bfd_proto_slow_timer_unset

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

Returns:

0 on success, otherwise one of the following error codes
BFD_API_INSTANCE_NOT_FOUND
BFD_API_SET_ERROR

2.1.2.63 s_int32_t smi_bfd_sess_gtsm_ttl_sdkapi (struct smiclient_globals * *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t * *sess_gtsm_ttl*, bool_t *snmp*)

This function is used to get the session GTSM(Generalized TTL Security Mechanism) TTL(Time to live) which specifies the minimum allowed TTL for received BFD control packets.GTSM TTL valid only when GTSM is enabled. smi_bfd_sess_gtsm_ttl_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

→ ***sess_gtsm_ttl*** GTSM TTL value numeric <0-255>

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.64 s_int32_t smi_bfd_set_sess_addr_type_sdkapi (struct smiclient_globals * *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t *sess_addr_type*, bool_t *snmp*)

This function is used to get the session address type of the BFD session. smi_bfd_set_-sess_addr_type_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

← ***sess_addr_type*** Session address type numeric <0-4>
0 - ADDR_TYPE_unknown
1 - ADDR_TYPE_ipv4
2 - ADDR_TYPE_ipv6
3 - ADDR_TYPE_ipv4z
4 - ADDR_TYPE_ipv6ze

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.65 s_int32_t smi_bfd_set_sess_admin_status_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t *set_sess_admin_status*, bool_t *snmp*)

This function is used to set the session administration status. smi_bfd_set_sess_-admin_status_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

← ***set_sess_admin_status*** Session administration status

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.66 s_int32_t smi_bfd_set_sess_detect_mult_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t *set_sess_detect_mult*, bool_t *snmp*)

This function is used to set the session detect time multiplier. smi_bfd_set_sess_-detect_mult_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

← ***set_sess_detect_mult*** The detect multiple value of the session

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.67 s_int32_t smi_bfd_set_sess_dmnd_mode_dsrd_flag_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t *sess_dmnd_mode_dsrd_flag*, bool_t *snmp*)

This function is used to set the session demand mode desired flag which indicates that the local system's desire to use Demand mode. smi_bfd_set_sess_dmnd_mode_dsrd_-flag_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

← ***sess_dmnd_mode_dsrd_flag*** Demand mode desired flag of the session

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.68 s_int32_t smi_bfd_set_sess_dsrd_min_tx_intvl_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t *sess_dsrd_min_tx_intvl*, bool_t *snmp*)

This function is used to set the session desired minimum transmission interval. smi_-bfd_set_sess_dsrd_min_tx_intvl_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

← ***sess_dsrd_min_tx_intvl*** Minimum transmission interval value of the session

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.69 s_int32_t smi_bfd_set_sess_echo_src_udp_port_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t *sess_echo_src_udp_port*, bool_t *snmp*)

This function is used to set the source UDP port number used for BFD session's echo packets. smi_bfd_set_sess_echo_src_udp_port_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

← ***sess_echo_src_udp_port*** Source UDP port number for echo packets

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.70 s_int32_t smi_bfd_set_sess_gtsm_sdkapi_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t *sess_gtsm*, bool_t *snmp*)

This function is used to set the session GTSM(Generalized TTL Security Mechanism) enable/disable mode. smi_bfd_set_sess_gtsm_sdkapi_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

← ***sess_gtsm*** Session address type numeric (1 | 2)
1 - BFD_GTSM_ENABLED
2 - BFD_GTSM_DISABLED

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.71 s_int32_t smi_bfd_set_sess_gtsm_ttl_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t *sess_gtsm_ttl*, bool_t *snmp*)

This function is used to set the session GTSM(Generalized TTL Security Mechanism) TTL(Time to live) which specifies the minimum allowed TTL for received BFD control packets. smi_bfd_set_sess_gtsm_ttl_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

← ***sess_gtsm_ttl*** GTSM TTL value numeric <0-255>

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.72 s_int32_t smi_bfd_set_sess_interface_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t *sess_interface*, bool_t *snmp*)

This function is used to get the session interface used to indicate the interface which the BFD session is running on. smi_bfd_set_sess_interface_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

← ***sess_interface*** BFD session interface

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.73 s_int32_t smi_bfd_set_sess_req_min_echo_rx_intvl_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t *sess_req_min_echo_rx_intvl*, bool_t *snmp*)

This function is used to set the session required minimum echo receive interval, which specifies the minimum interval, in microseconds, between received BFD Echo packets system is capable of supporting. smi_bfd_set_sess_req_min_echo_rx_intvl_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

← ***sess_req_min_echo_rx_intvl*** Required minimum echo receive interval of the session

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.74 s_int32_t smi_bfd_set_sess_req_min_rx_intvl_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t *sess_req_min_rx_intvl*, bool_t *snmp*)

This function is used to set the session required minimum receive interval, which specifies the minimum interval in microseconds. between received BFD Echo packets system is capable of supporting. smi_bfd_set_sess_req_min_rx_intvl_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

← ***sess_req_min_rx_intvl*** Required minimum receive interval of the session

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.75 s_int32_t smi_bfd_set_sess_row_status_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t *sess_row_status*, bool_t *snmp*)

This function is used to get the session row status. smi_bfd_set_sess_row_status_-sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

← ***sess_row_status*** Session row status

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.76 s_int32_t smi_bfd_set_sess_src_udp_port_sdkapi (struct smiclient_globals ∗ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t *sess_src_udp_port*, bool_t *snmp*)

This function is used to set the source UDP port number used for the BFD session's control packets. smi_bfd_set_sess_src_udp_port_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

← ***sess_src_udp_port*** Source UDP port number

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.77 s_int32_t smi_bfd_set_sess_stor_type_sdkapi (struct smiclient_globals $*$ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t *sess_stor_type*, bool_t *snmp*)

This function is used to set the session storage type. smi_bfd_set_sess_stor_type_-sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

← ***sess_stor_type*** Session storage type

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

2.1.2.78 s_int32_t smi_bfd_set_sess_version_no_sdkapi (struct smiclient_globals $*$ *azg*, u_int32_t *vr_id*, s_int32_t *proc_id*, u_int32_t *index*, u_int32_t *sess_version_no*, bool_t *snmp*)

This function is used to set the version number of the BFD protocol. smi_bfd_set_-sess_version_no_sdkapi

Parameters:

← ***azg*** Pointer to the SMI client global structure

← ***vr_id*** Virtual router ID.Default is 0, Pass 0 for a non-VR implementation

← ***proc_id*** Process ID for the BFD instance numeric <0-65535>

← ***index*** BFD session index value

← ***sess_version_no*** Version number

← ***snmp*** SNMP Enable/Disable numeric (0 | 1)
0 - PAL_FALSE (SNMP disabled)
1 - PAL_TRUE (SNMP enabled)

Returns:

BFD_SUCCESS on success, otherwise BFD_FAILURE when the function fails

Index