### ZebOS-XP BFD SMI Reference

IP Infusion Inc.

Generated by Doxygen 1.6.1

Wed Dec 16 12:33:43 2015

## **Contents**

1	File	Index			1
	1.1	File Li	st		1
2	File	Docum	entation		3
	2.1	smi_oa	am_bfd.h F	File Reference	3
		2.1.1	Detailed	Description	12
		2.1.2	Function	Documentation	12
			2.1.2.1	smi_bfd_add_user_session	12
			2.1.2.2	smi_bfd_api_get_sess_diag_sdkapi	13
			2.1.2.3	smi_bfd_api_sess_auth_key_id_sdkapi	14
			2.1.2.4	smi_bfd_api_sess_auth_type_sdkapi	14
			2.1.2.5	smi_bfd_del_user_session	15
			2.1.2.6	smi_bfd_echo_interval_set	15
			2.1.2.7	smi_bfd_echo_interval_unset	16
			2.1.2.8	smi_bfd_echo_mode_set	16
			2.1.2.9	smi_bfd_echo_mode_unset	16
			2.1.2.10	smi_bfd_get_discmap_index_sdkapi	17
			2.1.2.11	smi_bfd_get_perf_disc_time_sdkapi	17
			2.1.2.12	smi_bfd_get_perf_lastcomm_lost_diag_sdkapi	18
			2.1.2.13	smi_bfd_get_perf_lastses_down_time_sdkapi	18
			2.1.2.14	smi_bfd_get_perf_pkt_in_hc_sdkapi	19
			2.1.2.15	smi_bfd_get_perf_pkt_in_sdkapi	19
			2.1.2.16	smi_bfd_get_perf_pkt_out_hc_sdkapi	20
			2.1.2.17	smi_bfd_get_perf_pkt_out_sdkapi	20
			2.1.2.18	smi_bfd_get_perf_sess_up_count_sdkapi	21
			2 1 2 10	smi hfd get sess addr type sdkani	21

ii CONTENTS

2.1.2.20	smi_bfd_get_sess_admin_status_sdkapi	22
2.1.2.21	smi_bfd_get_sess_auth_pres_flag_sdkapi	23
2.1.2.22	$smi\_bfd\_get\_sess\_cntrlplane\_indep\_flag\_sdkapi \ . \ .$	23
2.1.2.23	smi_bfd_get_sess_dest_udp_port_sdkapi	24
2.1.2.24	smi_bfd_get_sess_detectmult_sdkapi	24
2.1.2.25	smi_bfd_get_sess_disc_sdkapi	25
2.1.2.26	$smi\_bfd\_get\_sess\_dmnd\_mode\_dsrd\_flag\_sdkapi \ .$	25
2.1.2.27	smi_bfd_get_sess_dsrd_min_tx_intvl_sdkapi	26
2.1.2.28	smi_bfd_get_sess_echo_src_udp_port_sdkapi	26
2.1.2.29	smi_bfd_get_sess_gtsm_sdkapi	27
2.1.2.30	smi_bfd_get_sess_interface_sdkapi	27
2.1.2.31	smi_bfd_get_sess_mh_unlnk_mode_sdkapi	28
2.1.2.32	smi_bfd_get_sess_neg_detect_mult_sdkapi	28
2.1.2.33	smi_bfd_get_sess_neg_echo_intvl_sdkapi	29
2.1.2.34	smi_bfd_get_sess_neg_intvl_sdkapi	29
2.1.2.35	smi_bfd_get_sess_oper_mode_sdkapi	30
2.1.2.36	$smi\_bfd\_get\_sess\_req\_min\_echo\_rx\_intvl\_sdkapi \ .$	30
2.1.2.37	smi_bfd_get_sess_req_min_rx_intvl_sdkapi	31
2.1.2.38	smi_bfd_get_sess_rmte_disc_sdkapi	31
2.1.2.39	smi_bfd_get_sess_rmte_heard_flag_sdkapi	32
2.1.2.40	smi_bfd_get_sess_row_status_sdkapi	32
2.1.2.41	smi_bfd_get_sess_src_udp_port_sdkapi	33
2.1.2.42	smi_bfd_get_sess_state_sdkapi	33
2.1.2.43	smi_bfd_get_sess_stor_type_sdkapi	34
2.1.2.44	smi_bfd_get_sess_type_sdkapi	34
2.1.2.45	smi_bfd_get_sess_up_time_sdkapi	35
2.1.2.46	smi_bfd_get_sess_version_sdkapi	35
2.1.2.47	smi_bfd_interface_echo_mode_set_sdkapi	36
2.1.2.48	smi_bfd_interface_echo_mode_unset_sdkapi	36
2.1.2.49	smi_bfd_interface_slow_timer_set_sdkapi	37
2.1.2.50	smi_bfd_interface_slow_timer_unset_sdkapi	37
2.1.2.51	smi_bfd_multihop_proto_interval_set_sdkapi	38
2.1.2.52	$smi\_bfd\_multihop\_proto\_interval\_unset\_sdkapi \ . \ .$	38
2.1.2.53	smi_bfd_notification_set	39

CONTENTS iii

2.1.2.54	smi_bfd_process_set	39
2.1.2.55	smi_bfd_proto_auth_set	40
2.1.2.56	smi_bfd_proto_auth_unset	41
2.1.2.57	smi_bfd_proto_interval_set_sdkapi	41
2.1.2.58	smi_bfd_proto_interval_unset_sdkapi	42
2.1.2.59	$smi\_bfd\_proto\_multihop\_auth\_ipv4\_set\_sdkapi  . \  \ .$	42
2.1.2.60	$smi\_bfd\_proto\_multihop\_auth\_unset\_ipv4\_sdkapi\ .$	43
2.1.2.61	smi_bfd_proto_slow_timer_set	44
2.1.2.62	smi_bfd_proto_slow_timer_unset	44
2.1.2.63	smi_bfd_sess_gtsm_ttl_sdkapi	45
2.1.2.64	smi_bfd_set_sess_addr_type_sdkapi	45
2.1.2.65	smi_bfd_set_sess_admin_status_sdkapi	46
2.1.2.66	smi_bfd_set_sess_detect_mult_sdkapi	46
2.1.2.67	$smi\_bfd\_set\_sess\_dmnd\_mode\_dsrd\_flag\_sdkapi \ .$	47
2.1.2.68	smi_bfd_set_sess_dsrd_min_tx_intvl_sdkapi	47
2.1.2.69	smi_bfd_set_sess_echo_src_udp_port_sdkapi	48
2.1.2.70	smi_bfd_set_sess_gtsm_sdkapi_sdkapi	48
2.1.2.71	smi_bfd_set_sess_gtsm_ttl_sdkapi	49
2.1.2.72	smi_bfd_set_sess_interface_sdkapi	49
2.1.2.73	smi_bfd_set_sess_req_min_echo_rx_intvl_sdkapi .	50
2.1.2.74	smi_bfd_set_sess_req_min_rx_intvl_sdkapi	50
2.1.2.75	smi_bfd_set_sess_row_status_sdkapi	51
2.1.2.76	smi_bfd_set_sess_src_udp_port_sdkapi	51
2.1.2.77	smi_bfd_set_sess_stor_type_sdkapi	52
2.1.2.78	smi_bfd_set_sess_version_no_sdkapi	52

## **Chapter 1**

## **File Index**

### 1.1 File List

Here is a list of all documented files with brief descriptions:	
smi_oam_bfd.h (Provides APIs for managing Bidirectional Forwarding De-	
tection(RFD) in ZebOS )	

2 File Index

### **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
- $\leftarrow$  *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
- $\rightarrow$  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
- $\leftarrow$  *snmp* SNMP Enable/Disable numeric (0 | 1)
  - 0 PAL\_FALSE (SNMP disabled)
  - 1 PAL\_TRUE (SNMP enabled)

#### **Returns:**

```
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
- $\leftarrow$  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

# 2.1.2.4 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
- $\leftarrow$  snmp SNMP Enable/Disable numeric (0 | 1)
  - 0 PAL\_FALSE (SNMP disabled)
  - 1 PAL\_TRUE (SNMP enabled)

#### **Returns:**

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>
- $\leftarrow$  *src* addr Source address for the session
- $\leftarrow$  dst addr Destination address for the session
- $\leftarrow$  *ifindex* Interface index
- $\leftarrow$  *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:**

- $\leftarrow$  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
- $\leftarrow$  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

2.1.2.11 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
- $\rightarrow \textit{perf\_lastcomm\_lost\_diag}$  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
- $\leftarrow$  *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

## 2.1.2.13 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>
- $\leftarrow$  *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:**

- $\leftarrow$  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
- $\leftarrow$  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

# 2.1.2.15 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
- $\leftarrow$  *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

```
2.1.2.16 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
- $\leftarrow$  *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

# 2.1.2.17 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
- $\leftarrow$  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

```
2.1.2.18 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
- $\leftarrow$  *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>
- $\leftarrow$  index BFD session index value
- → sess\_admin\_status Session administration status
- $\leftarrow$  *snmp* SNMP Enable/Disable numeric  $<0 \mid 1>$ 
  - 0 PAL\_FALSE (SNMP disabled)
  - 1 PAL\_TRUE (SNMP enabled)

#### **Returns:**

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>
- $\leftarrow$  *index* BFD session index value
- $\rightarrow$  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)
- $\leftarrow$  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

2.1.2.22 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
- $\rightarrow$  sess\_cntrlplane\_indep\_flag Control plane independent flag of the session numeric (0 | 1)
  - 0 PAL\_FALSE
  - 1 PAL\_TRUE
- $\leftarrow$  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

2.1.2.23 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
- $\leftarrow$  *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

2.1.2.24 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
- $\leftarrow$  snmp SNMP Enable/Disable numeric (0 | 1)
  - 0 PAL\_FALSE (SNMP disabled)
  - 1 PAL\_TRUE (SNMP enabled)

#### **Returns:**

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:**

- ← 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 disc Local session Discriminator
- $\leftarrow$  snmp SNMP Enable/Disable numeric  $<0 \mid 1>$ 
  - 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:**

- ← 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
- $\rightarrow$  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:**

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:**

- $\leftarrow$  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
- $\leftarrow$  snmp SNMP Enable/Disable numeric  $<0 \mid 1>$ 
  - 0 PAL\_FALSE (SNMP disabled)
  - 1 PAL\_TRUE (SNMP enabled)

#### **Returns:**

```
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
- $\rightarrow$  sess\_gtsm GTSM status numeric (0 | 1)
  - 0 PAL\_FALSE (GTSM disabled) 1 PAL\_TRUE (GTSM enabled)
- $\leftarrow$  *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

```
2.1.2.30 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>
- $\leftarrow$  *index* BFD session index value
- → sess\_interface Session interface
- $\leftarrow$  *snmp* SNMP Enable/Disable numeric (0 | 1)
  - 0 PAL\_FALSE (SNMP disabled)
  - 1 PAL\_TRUE (SNMP enabled)

#### **Returns:**

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
- $\leftarrow$  *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

2.1.2.32 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>
- $\leftarrow$  *index* BFD session index value
- → sess\_neg\_detect\_mult Negotiated detect multiple value
- $\leftarrow$  snmp SNMP Enable/Disable numeric (0 | 1)
  - 0 PAL\_FALSE (SNMP disabled)
  - 1 PAL\_TRUE (SNMP enabled)

#### **Returns:**

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
- $\leftarrow$  *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

2.1.2.34 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
- $\leftarrow$  *snmp* SNMP Enable/Disable numeric (0 | 1)
  - 0 PAL\_FALSE (SNMP disabled)
  - 1 PAL\_TRUE (SNMP enabled)

#### **Returns:**

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
- $\rightarrow$  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
- $\leftarrow$  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

2.1.2.36 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
- $\leftarrow$  *snmp* SNMP Enable/Disable numeric (0 | 1)
  - 0 PAL\_FALSE (SNMP disabled)
  - 1 PAL\_TRUE (SNMP enabled)

#### **Returns:**

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:**

- ← 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
- $\leftarrow$  *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
- ← 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\_disc Session remote discriminator
- $\leftarrow$  *snmp* SNMP Enable/Disable numeric  $<0 \mid 1>$ 
  - 0 PAL\_FALSE (SNMP disabled)
  - 1 PAL\_TRUE (SNMP enabled)

### **Returns:**

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
- $\rightarrow$  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
- $\leftarrow \textit{vr\_id}$  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
- $\leftarrow$  *snmp* SNMP Enable/Disable numeric (0 | 1)
  - 0 PAL\_FALSE (SNMP disabled)
  - 1 PAL TRUE (SNMP enabled)

# **Returns:**

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
- $\leftarrow$  snmp SNMP Enable/Disable numeric  $<0 \mid 1>$ 
  - 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
- $\leftarrow$  *snmp* SNMP Enable/Disable numeric (0 | 1)
  - 0 PAL\_FALSE (SNMP disabled)
  - 1 PAL\_TRUE (SNMP enabled)

### **Returns:**

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
- $\leftarrow$  *snmp* SNMP Enable/Disable numeric  $<0 \mid 1>$ 
  - 0 PAL\_FALSE (SNMP disabled) 1 PAL\_TRUE (SNMP enabled)

#### **Returns:**

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:

- $\leftarrow$  azg Pointer to the SMI client global structure
- ← 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>
- ← *index* BFD session index value
- → *version* Version number
- $\leftarrow$  *snmp* SNMP Enable/Disable numeric  $<0 \mid 1>$ 
  - 0 PAL\_FALSE (SNMP disabled) 1 PAL\_TRUE (SNMP enabled)

#### Returns:

# 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
- $\leftarrow vr\_id$  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>
- ← *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
- $\leftarrow \textit{vr\_id}$  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

- $\leftarrow$  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:**

- $\leftarrow$  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:**

- $\leftarrow$  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>
- $\leftarrow$  *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

- ← 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

- $\leftarrow$  *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

- ← 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

- ← 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
- $\leftarrow$  *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
- $\leftarrow$  *snmp* SNMP Enable/Disable numeric (0 | 1)
  - 0 PAL\_FALSE (SNMP disabled)
  - 1 PAL\_TRUE (SNMP enabled)

# **Returns:**

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
- $\leftarrow$  *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

2.1.2.66 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>
- $\leftarrow$  *index* BFD session index value
- ← set\_sess\_detect\_mult The detect multiple value of the session
- $\leftarrow$  *snmp* SNMP Enable/Disable numeric (0 | 1)
  - 0 PAL\_FALSE (SNMP disabled)
  - 1 PAL\_TRUE (SNMP enabled)

# **Returns:**

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
- $\leftarrow$  *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

2.1.2.68 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:**

- $\leftarrow$  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
- $\leftarrow$  snmp SNMP Enable/Disable numeric (0 | 1)
  - 0 PAL\_FALSE (SNMP disabled)
  - 1 PAL\_TRUE (SNMP enabled)

# **Returns:**

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
- $\leftarrow$  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:**

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>
- $\leftarrow$  *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

2.1.2.72 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:**

- $\leftarrow$  azg Pointer to the SMI client global structure
- ← 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>
- ← *index* BFD session index value
- ← sess\_interface BFD session interface
- $\leftarrow$  snmp SNMP Enable/Disable numeric (0 | 1)
  - 0 PAL\_FALSE (SNMP disabled)
  - 1 PAL\_TRUE (SNMP enabled)

# Returns:

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
- $\leftarrow$  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

2.1.2.74 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:**

- $\leftarrow$  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
- $\leftarrow$  snmp SNMP Enable/Disable numeric (0 | 1)
  - 0 PAL\_FALSE (SNMP disabled)
  - 1 PAL\_TRUE (SNMP enabled)

# **Returns:**

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
- $\leftarrow$  *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

2.1.2.76 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>
- $\leftarrow$  *index* BFD session index value
- ← *sess\_src\_udp\_port* Source UDP port number
- $\leftarrow$  *snmp* SNMP Enable/Disable numeric (0 | 1)
  - 0 PAL\_FALSE (SNMP disabled)
  - 1 PAL\_TRUE (SNMP enabled)

# **Returns:**

# 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
- $\leftarrow$  *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

# 2.1.2.78 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:**

# Index

smi_bfd_add_user_session	smi_oam_bfd.h, 22
smi_oam_bfd.h, 12	smi_bfd_get_sess_auth_pres_flag_sdkapi
smi_bfd_api_get_sess_diag_sdkapi	smi_oam_bfd.h, 22
smi_oam_bfd.h, 13	smi_bfd_get_sess_cntrlplane_indep
smi_bfd_api_sess_auth_key_id_sdkapi	flag_sdkapi
smi_oam_bfd.h, 13	smi_oam_bfd.h, 23
smi_bfd_api_sess_auth_type_sdkapi	smi_bfd_get_sess_dest_udp_port_sdkapi
smi_oam_bfd.h, 14	smi_oam_bfd.h, 24
smi_bfd_del_user_session	smi_bfd_get_sess_detectmult_sdkapi
smi_oam_bfd.h, 14	smi_oam_bfd.h, 24
smi_bfd_echo_interval_set	smi_bfd_get_sess_disc_sdkapi
smi_oam_bfd.h, 15	smi_oam_bfd.h, 24
smi_bfd_echo_interval_unset	smi_bfd_get_sess_dmnd_mode_dsrd
smi_oam_bfd.h, 15	flag_sdkapi
smi_bfd_echo_mode_set	smi_oam_bfd.h, 25
smi_oam_bfd.h, 16	smi_bfd_get_sess_dsrd_min_tx_intvl
smi_bfd_echo_mode_unset	sdkapi
smi_oam_bfd.h, 16	smi_oam_bfd.h, 25
smi_bfd_get_discmap_index_sdkapi	smi_bfd_get_sess_echo_src_udp_port
smi_oam_bfd.h, 17	sdkapi
smi_bfd_get_perf_disc_time_sdkapi	smi_oam_bfd.h, 26
smi_oam_bfd.h, 17	smi_bfd_get_sess_gtsm_sdkapi
smi_bfd_get_perf_lastcomm_lost_diag	smi_oam_bfd.h, 26
sdkapi	smi_bfd_get_sess_interface_sdkapi
smi_oam_bfd.h, 18	smi_oam_bfd.h, 27
smi_bfd_get_perf_lastses_down_time	smi_bfd_get_sess_mh_unlnk_mode
sdkapi	sdkapi
smi_oam_bfd.h, 18	smi_oam_bfd.h, 27
smi_bfd_get_perf_pkt_in_hc_sdkapi	smi_bfd_get_sess_neg_detect_mult
smi_oam_bfd.h, 19	sdkapi
smi_bfd_get_perf_pkt_in_sdkapi	smi_oam_bfd.h, 28
smi_oam_bfd.h, 19	smi_bfd_get_sess_neg_echo_intvl
smi_bfd_get_perf_pkt_out_hc_sdkapi	sdkapi
smi_oam_bfd.h, 20	smi_oam_bfd.h, 28
smi_bfd_get_perf_pkt_out_sdkapi	smi_bfd_get_sess_neg_intvl_sdkapi
smi_oam_bfd.h, 20	smi_oam_bfd.h, 29
smi_bfd_get_perf_sess_up_count_sdkapi	smi_bfd_get_sess_oper_mode_sdkapi
smi_oam_bfd.h, 21	smi_oam_bfd.h, 29
smi_bfd_get_sess_addr_type_sdkapi	smi_bfd_get_sess_req_min_echo_rx
smi_oam_bfd.h, 21	intvl_sdkapi
smi_bfd_get_sess_admin_status_sdkapi	smi_oam_bfd.h, 30

54 INDEX

smi_bfd_get_sess_req_min_rx_intvl	smi_bfd_proto_interval_unset_sdkapi
sdkapi	smi_oam_bfd.h, 42
smi_oam_bfd.h, 30	smi_bfd_proto_multihop_auth_ipv4
smi_bfd_get_sess_rmte_disc_sdkapi	set_sdkapi
smi_oam_bfd.h, 31	smi_oam_bfd.h, 42
smi_bfd_get_sess_rmte_heard_flag	smi_bfd_proto_multihop_auth_unset
sdkapi	ipv4_sdkapi
smi_oam_bfd.h, 31	smi_oam_bfd.h, 43
smi_bfd_get_sess_row_status_sdkapi	smi_bfd_proto_slow_timer_set
smi_oam_bfd.h, 32	smi_oam_bfd.h, 44
smi_bfd_get_sess_src_udp_port_sdkapi	smi_bfd_proto_slow_timer_unset
smi_oam_bfd.h, 32	smi_oam_bfd.h, 44
smi_bfd_get_sess_state_sdkapi	smi_bfd_sess_gtsm_ttl_sdkapi
smi_oam_bfd.h, 33	smi_oam_bfd.h, 44
smi_bfd_get_sess_stor_type_sdkapi	smi_bfd_set_sess_addr_type_sdkapi
smi_oam_bfd.h, 33	smi_oam_bfd.h, 45
smi_bfd_get_sess_type_sdkapi	smi_bfd_set_sess_admin_status_sdkapi
smi_oam_bfd.h, 34	smi_oam_bfd.h, 45
smi_bfd_get_sess_up_time_sdkapi	smi_bfd_set_sess_detect_mult_sdkapi
smi_oam_bfd.h, 34	smi_oam_bfd.h, 46
smi_bfd_get_sess_version_sdkapi	smi_bfd_set_sess_dmnd_mode_dsrd
smi_oam_bfd.h, 35	flag_sdkapi
smi_bfd_interface_echo_mode_set	smi_oam_bfd.h, 46
sdkapi	smi_bfd_set_sess_dsrd_min_tx_intvl
smi_oam_bfd.h, 35	sdkapi
smi_bfd_interface_echo_mode_unset	smi_oam_bfd.h, 47
sdkapi	smi_bfd_set_sess_echo_src_udp_port_
smi_oam_bfd.h, 36	sdkapi
smi_bfd_interface_slow_timer_set	smi_oam_bfd.h, 47
sdkapi	smi_bfd_set_sess_gtsm_sdkapi_sdkapi
smi_oam_bfd.h, 36	smi_oam_bfd.h, 48
smi_bfd_interface_slow_timer_unset	smi_bfd_set_sess_gtsm_ttl_sdkapi
sdkapi	smi_oam_bfd.h, 48
smi_oam_bfd.h, 37	smi_bfd_set_sess_interface_sdkapi
smi_bfd_multihop_proto_interval_set	smi_oam_bfd.h, 49
sdkapi	smi_bfd_set_sess_req_min_echo_rx
smi_oam_bfd.h, 37	intvl_sdkapi
smi_bfd_multihop_proto_interval	smi_oam_bfd.h, 49
unset_sdkapi	smi_bfd_set_sess_req_min_rx_intvl
smi_oam_bfd.h, 38	sdkapi
smi_bfd_notification_set	smi_oam_bfd.h, 50
smi_oam_bfd.h, 39	smi_bfd_set_sess_row_status_sdkapi
smi_bfd_process_set	smi_oam_bfd.h, 50
smi_oam_bfd.h, 39	smi_bfd_set_sess_src_udp_port_sdkapi
smi_bfd_proto_auth_set	smi_oam_bfd.h, 51
smi_oam_bfd.h, 39	smi_bfd_set_sess_stor_type_sdkapi
smi_bfd_proto_auth_unset	smi_oam_bfd.h, 51
smi_oam_bfd.h, 41	smi_bfd_set_sess_version_no_sdkapi
smi_bfd_proto_interval_set_sdkapi	smi_oam_bfd.h, 52
smi_oam_bfd.h, 41	smi_oam_bfd.h, 3
	_ ·

INDEX 55

smi_bfd_add_user_session, 12	smi_bfd_get_sess_interface_sdkapi,
smi_bfd_api_get_sess_diag_sdkapi,	27
13	smi_bfd_get_sess_mh_unlnk
smi_bfd_api_sess_auth_key_id	mode_sdkapi, 27
sdkapi, 13	smi_bfd_get_sess_neg_detect
smi_bfd_api_sess_auth_type	mult_sdkapi, 28
sdkapi, 14	smi_bfd_get_sess_neg_echo_intvl
smi_bfd_del_user_session, 14	sdkapi, 28
smi_bfd_echo_interval_set, 15	smi_bfd_get_sess_neg_intvl_sdkapi
smi_bfd_echo_interval_unset, 15	29
smi_bfd_echo_mode_set, 16	smi_bfd_get_sess_oper_mode
smi_bfd_echo_mode_unset, 16	sdkapi, 29
smi_bfd_get_discmap_index	smi_bfd_get_sess_req_min_echo
sdkapi, 17	rx_intvl_sdkapi, 30
smi_bfd_get_perf_disc_time	smi_bfd_get_sess_req_min_rx
sdkapi, 17	intvl_sdkapi, 30
smi_bfd_get_perf_lastcomm_lost	smi_bfd_get_sess_rmte_disc
diag_sdkapi, 18	sdkapi, 31
smi_bfd_get_perf_lastses_down	smi_bfd_get_sess_rmte_heard
time_sdkapi, 18	flag_sdkapi, 31
smi_bfd_get_perf_pkt_in_hc	smi_bfd_get_sess_row_status
sdkapi, 19	sdkapi, 32
smi_bfd_get_perf_pkt_in_sdkapi, 19	smi_bfd_get_sess_src_udp_port
smi_bfd_get_perf_pkt_out_hc	sdkapi, 32
sdkapi, 20	smi_bfd_get_sess_state_sdkapi, 33
smi_bfd_get_perf_pkt_out_sdkapi,	smi_bfd_get_sess_stor_type_sdkapi,
20	33
smi_bfd_get_perf_sess_up_count	smi_bfd_get_sess_type_sdkapi, 34
sdkapi, 21	smi_bfd_get_sess_up_time_sdkapi,
smi_bfd_get_sess_addr_type	34
sdkapi, 21	smi_bfd_get_sess_version_sdkapi,
<del>-</del>	35
smi_bfd_get_sess_admin_status sdkapi, 22	smi_bfd_interface_echo_mode
=	set_sdkapi, 35
smi_bfd_get_sess_auth_pres_flag	smi_bfd_interface_echo_mode
sdkapi, 22	unset_sdkapi, 36
smi_bfd_get_sess_cntrlplane	smi_bfd_interface_slow_timer_set
indep_flag_sdkapi, 23	sdkapi, 36
smi_bfd_get_sess_dest_udp_port	smi_bfd_interface_slow_timer
sdkapi, 24	unset_sdkapi, 37
smi_bfd_get_sess_detectmult	smi_bfd_multihop_proto_interval
sdkapi, 24	set_sdkapi, 37
smi_bfd_get_sess_disc_sdkapi, 24	smi_bfd_multihop_proto_interval
smi_bfd_get_sess_dmnd_mode	unset_sdkapi, 38
dsrd_flag_sdkapi, 25	smi_bfd_notification_set, 39
smi_bfd_get_sess_dsrd_min_tx	smi_bfd_process_set, 39
intvl_sdkapi, 25	smi_bfd_proto_auth_set, 39
smi_bfd_get_sess_echo_src_udp	smi_bfd_proto_auth_unset, 41
port_sdkapi, 26	smi_bfd_proto_interval_set_sdkapi,
smi_bfd_get_sess_gtsm_sdkapi, 26	41

56 INDEX

```
smi_bfd_proto_interval_unset_-
    sdkapi, 42
smi_bfd_proto_multihop_auth_-
    ipv4_set_sdkapi, 42
smi_bfd_proto_multihop_auth_-
    unset_ipv4_sdkapi, 43
smi_bfd_proto_slow_timer_set, 44
smi_bfd_proto_slow_timer_unset,
smi_bfd_sess_gtsm_ttl_sdkapi, 44
smi_bfd_set_sess_addr_type_-
    sdkapi, 45
smi_bfd_set_sess_admin_status_-
    sdkapi, 45
smi_bfd_set_sess_detect_mult_-
    sdkapi, 46
smi_bfd_set_sess_dmnd_mode_-
    dsrd_flag_sdkapi, 46
smi_bfd_set_sess_dsrd_min_tx_-
    intvl sdkapi, 47
smi_bfd_set_sess_echo_src_udp_-
    port_sdkapi, 47
smi_bfd_set_sess_gtsm_sdkapi_-
    sdkapi, 48
smi_bfd_set_sess_gtsm_ttl_sdkapi,
    48
smi_bfd_set_sess_interface_sdkapi,
smi_bfd_set_sess_req_min_echo_-
    rx_intvl_sdkapi, 49
smi_bfd_set_sess_req_min_rx_-
    intvl_sdkapi, 50
smi_bfd_set_sess_row_status_-
    sdkapi, 50
smi_bfd_set_sess_src_udp_port_-
    sdkapi, 51
smi_bfd_set_sess_stor_type_sdkapi,
smi_bfd_set_sess_version_no_-
    sdkapi, 52
```