SMP API

Generated by Doxygen 1.8.13

Contents

Chapter 1

Module Documentation

1.1 Security Manager Protocol (SMP)

Modules

- Security Manager API

1.1.1 Detailed Description

1.1.2 Introduction

The SMP subsystem implements the security manager protocol. It contains two independent subsystems:

- The initiator (SMPI). SMPI implements the initiator features of the security manager protocol and has support for multiple simultaneous connections.
- The responder (SMPR). SMPR implements the responder features of the security manager protocol and has support for only one connection.

SMP also implements:

- The cryptographic toolbox, which uses AES. The interface to AES is asynchronous and abstracted through WSF.
- Functions to support data signing.

An SMP layer below SMPI and SMPR implements routing of received SMP protocol messages to either SMPI or SMPR.

When connected in the central role, the following configurations are possible:

- SMP Secure Connections Initiator
- SMP Legacy Initiator

When connected in the peripheral role, the following configurations are possible:

- SMP Secure Connections Responder
- SMP Legacy Responder

For full API see Security Manager API.

1.1.3 SMP Secure Connections Responder

1.1.3.1 Introduction

This is the SMP Responder role for a connected device in the peripheral role when the Secure Connections feature is enabled. A peripheral device may initiate pairing with an SMP Slave Security Request or by waiting for the central device to send an SMP Pairing Request. A Secure Connections enabled peripheral device prefers to use the previously mentioned feature to pair, but may fall back to Legacy behavior if configured to do so.

1.1.3.2 SMP Secure Connections Responder Behavior

Here is the state machine for SMP_SC_RSP_BEHAVIOR

1.1.4 SMP Legacy Responder

1.1.4.1 Introduction

This is the SMP Responder role for a connected device in the peripheral role when the Legacy SMP configuration is used. A peripheral device may initiate pairing with an SMP Slave Seucrity Request or by waiting for the central device to send a SMP Pairing Request.

1.1.4.2 SMP Legacy Responder Behavior

Here is the state machine for SMP_LEG_RSP_BEHAVIOR.

1.1.5 SMP Secure Connections Initiator

1.1.5.1 Introduction

This is the SMP Initiator role for a connected device in the central role when the Secure Connection feature is enabled. A central device may initiate pairing by sending an SMP Pairing Request or by waiting for the peripheral device to send an SMP Slave Security Request. A Secure Connection enabled central device prefers to use the previously mentioned feature to pair, but may fall back to Legacy behavior if configued to do so.

1.1.5.2 SMP Secure Connections Initiator Behavior

Here is the state machine for SMP_SC_INT_BEHAVIOR

1.1.6 SMP Legacy Initiator

1.1.6.1 Introduction

This is the SMP Initiator role for a connected device in the central role when the Legacy SMP configuration is used. A central device may initiate pairing with an SMP Pairing Request or by waiting for the peripheral device to send an SMP Slave Security Request.

1.1.6.2 SMP Legacy Initiator Behavior

Here is the state machine for SMP_LEG_INT_BEHAVIOR.

1.2 Security Manager API

Data Structures

- struct smpCfg_t

  Configurable parameters.
- struct smpDmPair_t

  Data type for SMP_MSG_API_PAIR_REQ and SMP_MSG_API_PAIR_RSP.
- struct smpDmAuthRsp_t

  Data type for SMP_MSG_API_AUTH_RSP.
- struct smpDmKeypress_t

  Data type for SMP_MSG_API_USER_KEYPRESS.
- struct smpDmSecurityReq_t

  Data type for SMP_MSG_API_SECURITY_REQ.
- union smpDmMsg_t

  Union SMP DM message data types.

Macros

- #define SMP_HDR_LEN 1

  PDU format.
- #define SMP_TIMEOUT 30

  Protocol timeout.
- #define SMP_OOB_LEN 16

  OOB Data length in bytes.
- #define SMP_PIN_LEN 3

  Passkey Pin lenght in bytes.

SMP Events

Events recognized and handled by the SMP state machine.

- enum {
  SMP_MSG_API_PAIR_REQ = 1,
  SMP_MSG_API_PAIR_RSP,
  SMP_MSG_API_CANCEL_REQ,
  SMP_MSG_API_AUTH_RSP,
  SMP_MSG_API_SECURITY_REQ,
  SMP_MSG_CMD_PKT,
  SMP_MSG_CMD_PAIRING_FAILED,
  SMP_MSG_DM_ENCRYPT_CMPL,
  SMP_MSG_DM_ENCRYPT_FAILED,
  SMP_MSG_DM_CONN_CLOSE,
  SMP_MSG_WSF_AES_CMPL,
  SMP_MSG_INT_SEND_NEXT_KEY,
  SMP_MSG_INT_MAX_ATTEMPTS,
  SMP_MSG_INT_PAIRING_CMPL,
  SMP_MSG_INT_RSP_TIMEOUT,
  SMP_MSG_INT_WI_TIMEOUT,
  SMP_MSG_INT_LESC,
  SMP_MSG_INT_LEGACY,

SMP_MSG_INT_JW_NC,
SMP_MSG_INT_PASSKEY,
SMP_MSG_INT_OOB,
SMP_MSG_API_USER_CONFIRM,
SMP_MSG_API_USER_KEYPRESS,
SMP_MSG_API_KEYPRESS_CMPL,
SMP_MSG_WSF_ECC_CMPL,
SMP_MSG_INT_PK_NEXT,
SMP_MSG_INT_PK_CMPL,
SMP_MSG_WSF_CMAC_CMPL,
SMP_MSG_DH_CHECK_FAILURE,
SMP_MSG_EARLY_CNF,
SMP_MSG_INT_CLEANUP,
SMP_NUM_MSGS }

Event handler messages for SMP state machines.

- enum { SMP_DB_SERVICE_IND = SMP_NUM_MSGS }

  Additional SMP messages.

SMP Initialization Functions

Legacy and Secure Connections initialization for Initiator and Responder roles.

- void SmpiInit (void)

  Initialize SMP initiator role.
- void SmprInit (void)

  Initialize SMP responder role.
- void SmpiScInit (void)

  Initialize SMP initiator role utilizing BTLE Secure Connections.
- void SmprScInit (void)

  Initialize SMP responder role utilizing BTLE Secure Connections.
- void SmpNonInit (void)

  Use this SMP init function when SMP is not supported.

SMP DM Interface Functions

Functions that allow the DM to send messages to SMP.

- void SmpDmMsgSend (smpDmMsg_t *pMsg)

  This function is called by DM to send a message to SMP.
- void SmpDmEncryptInd (wsfMsgHdr_t *pMsg)

  This function is called by DM to notify SMP of encrypted link status.
- bool_t SmpDmLescEnabled (dmConnId_t connId)

  Check if LE Secure Connections is enabled on the connection.
- uint8_t * SmpDmGetStk (dmConnId_t connId, uint8_t *pSecLevel)

  Return the STK for the given connection.
- void SmpScGetCancelMsgWithReattempt (dmConnId_t connId, wsfMsgHdr_t *pHdr, uint8_t status)

  Format a cancel message with consideration for the attempts counter.
- void SmpDbInit (void)

  Initialize the SMP Database.
- void SmpScEnableZeroDhKey (bool_t enable)

  Called to force the DhKey to zero for qualification test purposes.

SMP Encryption Key Size

- #define SMP_KEY_SIZE_MAX 16

  Maximum encryption key size.
- #define SMP_KEY_SIZE_MIN 7

  Minimum encryption key size.

SMP Error Codes

SMP PDU status codes

- #define SMP_ERR_PASSKEY_ENTRY 0x01

  User input of passkey failed.
- #define SMP_ERR_OOB 0x02

  OOB data is not available.
- #define SMP_ERR_AUTH_REQ 0x03

  Authentication requirements cannot be met.
- #define SMP_ERR_CONFIRM_VALUE 0x04

  Confirm value does not match.
- #define SMP_ERR_PAIRING_NOT_SUP 0x05

  Pairing is not supported by the device.
- #define SMP_ERR_ENC_KEY_SIZE 0x06

  Insufficient encryption key size.
- #define SMP_ERR_COMMAND_NOT_SUP 0x07

  Command not supported.
- #define SMP_ERR_UNSPECIFIED 0x08

  Unspecified reason.
- #define SMP_ERR_ATTEMPTS 0x09

  Repeated attempts.
- #define SMP_ERR_INVALID_PARAM 0x0A

  Invalid parameter or command length.
- #define SMP_ERR_DH_KEY_CHECK 0x0B

  DH Key check did not match.
- #define SMP_ERR_NUMERIC_COMPARISON 0x0C

  Numeric comparison did not match.
- #define SMP_ERR_BR_EDR_IN_PROGRESS 0x0D

  BR/EDR in progress.
- #define SMP_ERR_CROSS_TRANSPORT 0x0E

  BR/EDR cross transport key generation not allowed.

Proprietary Error Codes

Internal error codes not sent in any SMP PDU.

- #define SMP_ERR_MEMORY 0xE0

  Out of memory.
- #define SMP_ERR_TIMEOUT 0xE1

  Transaction timeout.

SMP PDU Codes

SMP PDU Code describing command received or sent.

- #define SMP_CMD_PAIR_REQ 0x01
  Pairing request.
- #define SMP_CMD_PAIR_RSP 0x02
  Pairing response.
- #define SMP_CMD_PAIR_CNF 0x03
  Pairing confirm.
- #define SMP_CMD_PAIR_RAND 0x04
  Pairing random.
- #define SMP_CMD_PAIR_FAIL 0x05
  Pairing failed.
- #define SMP_CMD_ENC_INFO 0x06
  Encryption information.
- #define SMP_CMD_MASTER_ID 0x07
  Master identification.
- #define SMP_CMD_ID_INFO 0x08
  Identity information.
- #define SMP_CMD_ID_ADDR_INFO 0x09
  Identity address information.
- #define SMP_CMD_SIGN_INFO 0x0A
  Signing information.
- #define SMP_CMD_SECURITY_REQ 0x0B
  Security fequest.
- #define SMP_CMD_PUBLIC_KEY 0x0C
  Public Kkey.
- #define SMP_CMD_DHKEY_CHECK 0x0D
  DH Key check.
- #define SMP_CMD_KEYPRESS 0x0E
  User key press.
- #define SMP_CMD_MAX 0x0F
  Command code maximum.

SMP PDU Packet Lengths

Fixed length of the PDU to be sent.

- #define SMP_PAIR_REQ_LEN 7
  Pairing request message length.
- #define SMP_PAIR_RSP_LEN 7
  Pairing response message length.
- #define SMP_PAIR_CNF_LEN 17
  Pairing confirmation message length.
- #define SMP_PAIR_RAND_LEN 17
  Pairing random message length.
- #define SMP_PAIR_FAIL_LEN 2
  Pairing fail message length.

- #define SMP_ENC_INFO_LEN 17

  Encryption information message length.
- #define SMP_MASTER_ID_LEN 11

  Master identification messagelength.
- #define SMP_ID_INFO_LEN 17

  Identity information message length.
- #define SMP_ID_ADDR_INFO_LEN 8

  Identity address information message length.
- #define SMP_SIGN_INFO_LEN 17

  Signing information message length.
- #define SMP_SECURITY_REQ_LEN 2

  Security request message length.
- #define SMP_PUB_KEY_MSG_LEN (1 + 2∗SMP_PUB_KEY_LEN)

  Public key message length.
- #define SMP_DHKEY_CHECK_MSG_LEN (1 + SMP_DHKEY_CHECK_LEN)

  Diffie-Hellman key check message length.
- #define SMP_KEYPRESS_MSG_LEN 2

  Keypress message length.

SMP I/O Capabilities

I/O capabilities codes to be set for SMP_CMD_PAIR_REQ and SMP_CMD_PAIR_RSP

- #define SMP_IO_DISP_ONLY 0x00

  Display only.
- #define SMP_IO_DISP_YES_NO 0x01

  Display yes/no.
- #define SMP_IO_KEY_ONLY 0x02

  Keyboard only.
- #define SMP_IO_NO_IN_NO_OUT 0x03

  No input, no output.
- #define SMP_IO_KEY_DISP 0x04

  Keyboard display.

SMP OOB Data Flag

Out-of-Band codes to be set for SMP_CMD_PAIR_REQ and SMP_CMD_PAIR_RSP

- #define SMP_OOB_DATA_NONE 0x00

  No OOB data from the remote device is present.
- #define SMP_OOB_DATA_PRESENT 0x01

  OOB data from the remote device is present.

SMP Authentication Requirements Flags

Authentication Requirements Flags to be set for SMP_CMD_PAIR_REQ and SMP_CMD_PAIR_RSP.

- #define SMP_AUTH_BOND_MASK 0x03
  Mask for bonding bits.
- #define SMP_AUTH_BOND_FLAG 0x01
  Bonding requested.
- #define SMP_AUTH_MITM_FLAG 0x04
  MITM (authenticated pairing) requested.
- #define SMP_AUTH_SC_FLAG 0x08
  LE Secure Connections requested.
- #define SMP_AUTH_KP_FLAG 0x10
  Keypress notifications requested.

SMP Key Distribution Flags

Flags of security keys this device is requesting to be distribution once pairing completes.

- #define SMP_KEY_DIST_ENC 0x01
  Distribute LTK.
- #define SMP_KEY_DIST_ID 0x02
  Distribute IRK.
- #define SMP_KEY_DIST_SIGN 0x04
  Distribute CSRK.
- #define SMP_KEY_DIST_MASK (SMP_KEY_DIST_ENC | SMP_KEY_DIST_ID | SMP_KEY_DIST_SIGN)
  Key distribution mask.

SMP LE Secure Connection Keypress Codes

Keypress codes found in SMP_CMD_KEYPRESS PDU to be sent on the respective action when the SMP_AUTH_KP_FLAG is set in both the SMP_CMD_PAIR_REQ and SMP_CMD_PAIR_RSP.

- #define SMP_PASSKEY_ENTRY_STARTED 0x00
  Passkey entry started keypress type.
- #define SMP_PASSKEY_DIGIT_ENTERED 0x01
  Passkey digit entered keypress type.
- #define SMP_PASSKEY_DIGIT_ERASED 0x02
  Passkey digit erased keypress type.
- #define SMP_PASSKEY_CLEARED 0x03
  Passkey cleared keypress type.
- #define SMP_PASSKEY_ENTRY_COMPLETED 0x04
  Passkey entry complete keypress type.

SMP Value Length Constants

Lengths of various keys and values.

- #define SMP_RAND_LEN 16

  Random number length.
- #define SMP_CONFIRM_LEN 16

  Confirm number length.
- #define SMP_KEY_LEN 16

  Key length.
- #define SMP_RAND8_LEN 8

  Random 8-byte number length.
- #define SMP_PRIVATE_KEY_LEN 32

  Secure connections private key length.
- #define SMP_PUB_KEY_LEN 32

  Secure connecdtions public key length.
- #define SMP_DHKEY_LEN 32

  Secure connection Diffie-Hellman key length.
- #define SMP_DHKEY_CHECK_LEN 16

  Secure connection Diffie-Hellman key check length.

CMAC Input Lengths Constants

Input lengths of SMP cryptopgraphic toolbox functions.

- #define SMP_F4_TEXT_LEN (SMP_PUB_KEY_LEN ∗ 2 + 1)

  F4 input length.
- #define SMP_G2_TEXT_LEN (SMP_PUB_KEY_LEN ∗ 2 + SMP_RAND_LEN)

  G2 input length.
- #define SMP_F5_TKEY_TEXT_LEN (SMP_DHKEY_LEN)

  F5 Temporary key input length.
- #define SMP_F5_TEXT_LEN (9 + 2∗BDA_ADDR_LEN + 2∗SMP_RAND_LEN)

  F5 input length.
- #define SMP_F6_TEXT_LEN (2∗BDA_ADDR_LEN + 3∗SMP_RAND_LEN + 5)

  F6 input length.

1.2.1 Detailed Description

1.2.2 Macro Definition Documentation

1.2.2.1 SMP_HDR_LEN

```
#define SMP_HDR_LEN 1
```

PDU format.

Attribute PDU header length.

Definition at line 39 of file smp_defs.h.

1.2.2.2 SMP_TIMEOUT

```
#define SMP_TIMEOUT 30
```

Protocol timeout.

Protocol timeout in seconds.

Definition at line 42 of file smp_defs.h.

1.2.3 Enumeration Type Documentation

1.2.3.1 anonymous enum

```
anonymous enum
```

Event handler messages for SMP state machines.

Enumerator

SMP_MSG_API_PAIR_REQ	API pairing request.
SMP_MSG_API_PAIR_RSP	API pairing response.
SMP_MSG_API_CANCEL_REQ	API cancel request.
SMP_MSG_API_AUTH_RSP	API pin response.
SMP_MSG_API_SECURITY_REQ	API security request.
SMP_MSG_CMD_PKT	SMP command packet received.
SMP_MSG_CMD_PAIRING_FAILED	SMP pairing failed packet received.
SMP_MSG_DM_ENCRYPT_CMPL	Link encrypted.
SMP_MSG_DM_ENCRYPT_FAILED	Link encryption failed.
SMP_MSG_DM_CONN_CLOSE	Connection closed.
SMP_MSG_WSF_AES_CMPL	AES calculation complete.
SMP_MSG_INT_SEND_NEXT_KEY	Send next key to be distributed.
SMP_MSG_INT_MAX_ATTEMPTS	Maximum pairing attempts reached.
SMP_MSG_INT_PAIRING_CMPL	Pairing complete.
SMP_MSG_INT_RSP_TIMEOUT	Pairing protocol response timeout.
SMP_MSG_INT_WI_TIMEOUT	Pairing protocol wait interval timeout.

Enumerator

SMP_MSG_INT_LESC	Pair with Secure Connections.
SMP_MSG_INT_LEGACY	Pair with Legacy Security.
SMP_MSG_INT_JW_NC	LESC Just-Works/Numeric Comparison pairing.
SMP_MSG_INT_PASSKEY	LESC Passkey pairing.
SMP_MSG_INT_OOB	LESC Out-of-Band Pairing.
SMP_MSG_API_USER_CONFIRM	User confirms valid numeric comparison.
SMP_MSG_API_USER_KEYPRESS	User keypress in passkey pairing.
SMP_MSG_API_KEYPRESS_CMPL	User keypress complete in passkey pairing.
SMP_MSG_WSF_ECC_CMPL	WSF ECC operation complete.
SMP_MSG_INT_PK_NEXT	Continue to next passkey bit.
SMP_MSG_INT_PK_CMPL	Passkey operation complete.
SMP_MSG_WSF_CMAC_CMPL	WSF CMAC operation complete.
SMP_MSG_DH_CHECK_FAILURE	DHKey check failure.
SMP_MSG_EARLY_CNF	An early Confirm from the initiator in passkey pairing.
SMP_MSG_INT_CLEANUP	Cleanup control information and return to IDLE state.
SMP_NUM_MSGS	Number of SMP message types.

Definition at line 72 of file smp_api.h.

```
73 {
74   SMP_MSG_API_PAIR_REQ = 1,              /*!< \brief API pairing request */
75   SMP_MSG_API_PAIR_RSP,                  /*!< \brief API pairing response */
76   SMP_MSG_API_CANCEL_REQ,                /*!< \brief API cancel request */
77   SMP_MSG_API_AUTH_RSP,                  /*!< \brief API pin response */
78   SMP_MSG_API_SECURITY_REQ,              /*!< \brief API security request */
79   SMP_MSG_CMD_PKT,                       /*!< \brief SMP command packet received */
80   SMP_MSG_CMD_PAIRING_FAILED,            /*!< \brief SMP pairing failed packet
       received */
81   SMP_MSG_DM_ENCRYPT_CMPL,               /*!< \brief Link encrypted */
82   SMP_MSG_DM_ENCRYPT_FAILED,             /*!< \brief Link encryption failed */
83   SMP_MSG_DM_CONN_CLOSE,                 /*!< \brief Connection closed */
84   SMP_MSG_WSF_AES_CMPL,                  /*!< \brief AES calculation complete */
85   SMP_MSG_INT_SEND_NEXT_KEY,             /*!< \brief Send next key to be
       distributed */
86   SMP_MSG_INT_MAX_ATTEMPTS,              /*!< \brief Maximum pairing attempts
       reached */
87   SMP_MSG_INT_PAIRING_CMPL,              /*!< \brief Pairing complete */
88   SMP_MSG_INT_RSP_TIMEOUT,               /*!< \brief Pairing protocol response
       timeout */
89   SMP_MSG_INT_WI_TIMEOUT,                /*!< \brief Pairing protocol wait interval
       timeout */
90   SMP_MSG_INT_LESC,                      /*!< \brief Pair with Secure Connections */
91   SMP_MSG_INT_LEGACY,                    /*!< \brief Pair with Legacy Security */
92   SMP_MSG_INT_JW_NC,                     /*!< \brief LESC Just-Works/Numeric Comparison
       pairing */
93   SMP_MSG_INT_PASSKEY,                   /*!< \brief LESC Passkey pairing */
94   SMP_MSG_INT_OOB,                       /*!< \brief LESC Out-of-Band Pairing */
95   SMP_MSG_API_USER_CONFIRM,              /*!< \brief User confirms valid numeric
       comparison */
96   SMP_MSG_API_USER_KEYPRESS,             /*!< \brief User keypress in passkey
       pairing */
97   SMP_MSG_API_KEYPRESS_CMPL,             /*!< \brief User keypress complete in
       passkey pairing */
98   SMP_MSG_WSF_ECC_CMPL,                  /*!< \brief WSF ECC operation complete */
99   SMP_MSG_INT_PK_NEXT,                   /*!< \brief Continue to next passkey bit */
100   SMP_MSG_INT_PK_CMPL,                   /*!< \brief Passkey operation complete */
101   SMP_MSG_WSF_CMAC_CMPL,                 /*!< \brief WSF CMAC operation complete */
102   SMP_MSG_DH_CHECK_FAILURE,              /*!< \brief DHKey check failure */
103   SMP_MSG_EARLY_CNF,                     /*!< \brief An early Confirm from the initiator
       in passkey pairing */
104   SMP_MSG_INT_CLEANUP,                   /*!< \brief Cleanup control information and
       return to IDLE state */
105   SMP_NUM_MSGS                           /*!< \brief Number of SMP message types. */
106 };
```

1.2.3.2 anonymous enum

```
anonymous enum
```

Additional SMP messages.

Enumerator

SMP_DB_SERVICE_IND	SMP DB Service timer indication.

Definition at line 111 of file smp_api.h.

```
112 {
113   SMP_DB_SERVICE_IND = SMP_NUM_MSGS       /*!< \brief SMP DB Service timer
       indication */
114 };
```

1.2.4 Function Documentation

1.2.4.1 SmpiInit()

```
void SmpiInit (
              void  )
```

Initialize SMP initiator role.

Returns

None.

1.2.4.2 SmprInit()

```
void SmprInit (
              void  )
```

Initialize SMP responder role.

Returns

None.

1.2.4.3 SmpiScInit()

```
void SmpiScInit (
             void  )
```

Initialize SMP initiator role utilizing BTLE Secure Connections.

Returns

None.

1.2.4.4 SmprScInit()

```
void SmprScInit (
             void  )
```

Initialize SMP responder role utilizing BTLE Secure Connections.

Returns

None.

1.2.4.5 SmpNonInit()

```
void SmpNonInit (
             void  )
```

Use this SMP init function when SMP is not supported.

Returns

None.

1.2.4.6 SmpDmMsgSend()

```
void SmpDmMsgSend (
             smpDmMsg_t * pMsg )
```

This function is called by DM to send a message to SMP.

Parameters

pMsg	Pointer to message structure.

Returns

None.

1.2.4.7 SmpDmEncryptInd()

```
void SmpDmEncryptInd (
             wsfMsgHdr_t * pMsg )
```

This function is called by DM to notify SMP of encrypted link status.

Parameters

pMsg	Pointer to HCI message structure.

Returns

None.

1.2.4.8 SmpDmLescEnabled()

```
bool_t SmpDmLescEnabled (
             dmConnId_t connId )
```

Check if LE Secure Connections is enabled on the connection.

Parameters

connId	Connection identifier.

Returns

TRUE is Secure Connections is enabled, else FALSE

1.2.4.9 SmpDmGetStk()

```
uint8_t* SmpDmGetStk (
             dmConnId_t connId,
             uint8_t * pSecLevel )
```

Return the STK for the given connection.

Parameters

connId	Connection identifier.
pSecLevel	Returns the security level of pairing when STK was created.

Returns

Pointer to STK or NULL if not available.

1.2.4.10 SmpScGetCancelMsgWithReattempt()

```
void SmpScGetCancelMsgWithReattempt (
             dmConnId_t connId,
             wsfMsgHdr_t * pHdr,
             uint8_t status )
```

Format a cancel message with consideration for the attempts counter.

Parameters

connId	Connection Id.
pHdr	Pointer to header of message to fill.
status	Status to include.

Returns

none.

1.2.4.11 SmpDbInit()

```
void SmpDbInit (
             void  )
```

Initialize the SMP Database.

Returns

None.

1.2.4.12 SmpScEnableZeroDhKey()

```
void SmpScEnableZeroDhKey (
             bool_t enable )
```

Called to force the DhKey to zero for qualification test purposes.

Parameters

enable	TRUE - Force DhKey to zero. FALSE - Use calculated key

Returns

None.

1.3 STACK_INIT

SMP Configuration Structure

Pointer to structure containing initialization details of the SMP Subsystem. To be configured by Application.

- smpCfg_t * pSmpCfg
  Configuration pointer.

1.3.1 Detailed Description

1.4 STACK_EVENT

SMP Event Handling

Message passing interface to SMP from other tasks through WSF.

- void SmpHandlerInit (wsfHandlerId_t handlerId)

  SMP handler init function called during system initialization.
- void SmpHandler (wsfEventMask_t event, wsfMsgHdr_t *pMsg)

  WSF event handler for SMP.

1.4.1 Detailed Description

1.4.2 Function Documentation

1.4.2.1 SmpHandlerInit()

```
void SmpHandlerInit (
            wsfHandlerId_t handlerId )
```

SMP handler init function called during system initialization.

Parameters

handlerId	WSF handler ID for SMP.

Returns

None.

1.4.2.2 SmpHandler()

```
void SmpHandler (
            wsfEventMask_t event,
            wsfMsgHdr_t * pMsg )
```

WSF event handler for SMP.

Parameters

event	WSF event mask.
pMsg	WSF message.

Returns

None.

Chapter 2

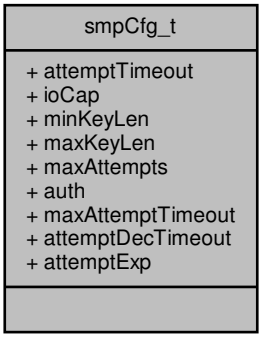
Data Structure Documentation

2.1 smpCfg_t Struct Reference

Configurable parameters.

`#include <smp_api.h>`

Collaboration diagram for smpCfg_t:



Data Fields

- uint32_t attemptTimeout

  'Repeated attempts' timeout in msec
- uint8_t ioCap

  I/O Capability.
- uint8_t minKeyLen

  Minimum encryption key length.
- uint8_t maxKeyLen

  Maximum encryption key length.

- uint8_t maxAttempts

  Attempts to trigger 'repeated attempts' timeout.
- uint8_t auth

  Device authentication requirements.
- uint32_t maxAttemptTimeout

  Maximum 'Repeated attempts' timeout in msec.
- uint32_t attemptDecTimeout

  Time msec before attemptExp decreases.
- uint16_t attemptExp

  Exponent to raise attemptTimeout on maxAttempts.

2.1.1 Detailed Description

Configurable parameters.

Definition at line 122 of file smp_api.h.

The documentation for this struct was generated from the following file:

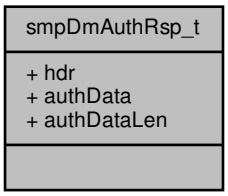
- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/ble-host/include/smp_api.h

2.2 smpDmAuthRsp_t Struct Reference

Data type for SMP_MSG_API_AUTH_RSP.

```
#include <smp_api.h>
```

Collaboration diagram for smpDmAuthRsp_t:



Data Fields

- wsfMsgHdr_t hdr

  Message header.
- uint8_t authData [SMP_OOB_LEN]

  Authentication data to display.
- uint8_t authDataLen

  Length of authentication data.

2.2.1 Detailed Description

Data type for SMP_MSG_API_AUTH_RSP.

Definition at line 146 of file smp_api.h.

The documentation for this struct was generated from the following file:

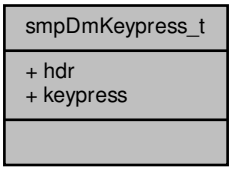
- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/ble-host/include/smp_api.h

2.3 smpDmKeypress_t Struct Reference

Data type for SMP_MSG_API_USER_KEYPRESS.

`#include <smp_api.h>`

Collaboration diagram for smpDmKeypress_t:



Data Fields

- wsfMsgHdr_t hdr
  Message header.
- uint8_t keypress
  Keypress.

2.3.1 Detailed Description

Data type for SMP_MSG_API_USER_KEYPRESS.

Definition at line 154 of file smp_api.h.

The documentation for this struct was generated from the following file:

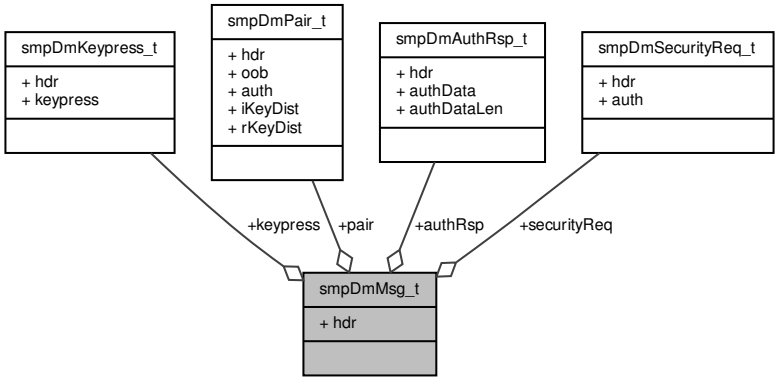
- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/ble-host/include/smp_api.h

2.4 smpDmMsg_t Union Reference

Union SMP DM message data types.

`#include <smp_api.h>`

Collaboration diagram for smpDmMsg_t:



Data Fields

- wsfMsgHdr_t hdr

  Message header.
- smpDmPair_t pair

  Pairing request/response message.
- smpDmAuthRsp_t authRsp

  Authentication message.
- smpDmSecurityReq_t securityReq

  Security Request message.
- smpDmKeypress_t keypress

  Keypress message.

2.4.1 Detailed Description

Union SMP DM message data types.

Definition at line 168 of file smp_api.h.

The documentation for this union was generated from the following file:

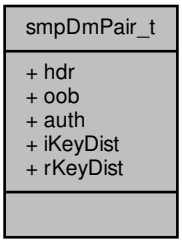
- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/ble-host/include/smp_api.h

2.5 smpDmPair_t Struct Reference

Data type for SMP_MSG_API_PAIR_REQ and SMP_MSG_API_PAIR_RSP.

`#include <smp_api.h>`

Collaboration diagram for smpDmPair_t:



Data Fields

- wsfMsgHdr_t hdr

  Message header.
- uint8_t oob

  Out-of-band data present flag.
- uint8_t auth

  authentication flags
- uint8_t iKeyDist

  Initiator key distribution flags.
- uint8_t rKeyDist

  Responder key distribution flags.

2.5.1 Detailed Description

Data type for SMP_MSG_API_PAIR_REQ and SMP_MSG_API_PAIR_RSP.

Definition at line 136 of file smp_api.h.

The documentation for this struct was generated from the following file:

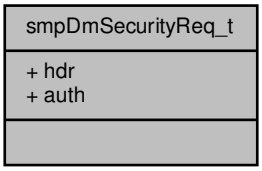
- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/ble-host/include/smp_api.h

2.6 smpDmSecurityReq_t Struct Reference

Data type for SMP_MSG_API_SECURITY_REQ.

`#include <smp_api.h>`

Collaboration diagram for smpDmSecurityReq_t:



Data Fields

- wsfMsgHdr_t hdr

  Message header.
- uint8_t auth

  Authentication flags.

2.6.1 Detailed Description

Data type for SMP_MSG_API_SECURITY_REQ.

Definition at line 161 of file smp_api.h.

The documentation for this struct was generated from the following file:

- /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/ble-host/include/smp_api.h

Chapter 3

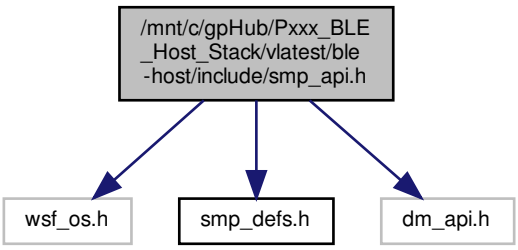
File Documentation

3.1 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/ble-host/include/smp_api.h File Reference

SMP subsystem API.

```
#include "wsf_os.h"
#include "smp_defs.h"
#include "dm_api.h"
```

Include dependency graph for smp_api.h:



Data Structures

- struct smpCfg_t

  Configurable parameters.
- struct smpDmPair_t

  Data type for SMP_MSG_API_PAIR_REQ and SMP_MSG_API_PAIR_RSP.
- struct smpDmAuthRsp_t

  Data type for SMP_MSG_API_AUTH_RSP.
- struct smpDmKeypress_t

  Data type for SMP_MSG_API_USER_KEYPRESS.
- struct smpDmSecurityReq_t

  Data type for SMP_MSG_API_SECURITY_REQ.
- union smpDmMsg_t

  Union SMP DM message data types.

Enumerations

SMP Events

Events recognized and handled by the SMP state machine.

- enum {
  SMP_MSG_API_PAIR_REQ = 1,
  SMP_MSG_API_PAIR_RSP,
  SMP_MSG_API_CANCEL_REQ,
  SMP_MSG_API_AUTH_RSP,
  SMP_MSG_API_SECURITY_REQ,
  SMP_MSG_CMD_PKT,
  SMP_MSG_CMD_PAIRING_FAILED,
  SMP_MSG_DM_ENCRYPT_CMPL,
  SMP_MSG_DM_ENCRYPT_FAILED,
  SMP_MSG_DM_CONN_CLOSE,
  SMP_MSG_WSF_AES_CMPL,
  SMP_MSG_INT_SEND_NEXT_KEY,
  SMP_MSG_INT_MAX_ATTEMPTS,
  SMP_MSG_INT_PAIRING_CMPL,
  SMP_MSG_INT_RSP_TIMEOUT,
  SMP_MSG_INT_WI_TIMEOUT,
  SMP_MSG_INT_LESC,
  SMP_MSG_INT_LEGACY,
  SMP_MSG_INT_JW_NC,
  SMP_MSG_INT_PASSKEY,
  SMP_MSG_INT_OOB,
  SMP_MSG_API_USER_CONFIRM,
  SMP_MSG_API_USER_KEYPRESS,
  SMP_MSG_API_KEYPRESS_CMPL,
  SMP_MSG_WSF_ECC_CMPL,
  SMP_MSG_INT_PK_NEXT,
  SMP_MSG_INT_PK_CMPL,
  SMP_MSG_WSF_CMAC_CMPL,
  SMP_MSG_DH_CHECK_FAILURE,
  SMP_MSG_EARLY_CNF,
  SMP_MSG_INT_CLEANUP,
  SMP_NUM_MSGS }

  Event handler messages for SMP state machines.

- enum { SMP_DB_SERVICE_IND = SMP_NUM_MSGS }

  Additional SMP messages.

Functions

SMP Initialization Functions

Legacy and Secure Connections initialization for Initiator and Responder roles.

- void SmpiInit (void)

  Initialize SMP initiator role.
- void SmprInit (void)

  Initialize SMP responder role.
- void SmpiScInit (void)

Initialize SMP initiator role utilizing BTLE Secure Connections.

- void SmprScInit (void)

  Initialize SMP responder role utilizing BTLE Secure Connections.
- void SmpNonInit (void)

  Use this SMP init function when SMP is not supported.

SMP DM Interface Functions

Functions that allow the DM to send messages to SMP.

- void SmpDmMsgSend (smpDmMsg_t *pMsg)

  This function is called by DM to send a message to SMP.
- void SmpDmEncryptInd (wsfMsgHdr_t *pMsg)

  This function is called by DM to notify SMP of encrypted link status.
- bool_t SmpDmLescEnabled (dmConnId_t connId)

  Check if LE Secure Connections is enabled on the connection.
- uint8_t * SmpDmGetStk (dmConnId_t connId, uint8_t *pSecLevel)

  Return the STK for the given connection.
- void SmpScGetCancelMsgWithReattempt (dmConnId_t connId, wsfMsgHdr_t *pHdr, uint8_t status)

  Format a cancel message with consideration for the attempts counter.
- void SmpDbInit (void)

  Initialize the SMP Database.
- void SmpScEnableZeroDhKey (bool_t enable)

  Called to force the DhKey to zero for qualification test purposes.

Variables

SMP Configuration Structure

Pointer to structure containing initialization details of the SMP Subsystem. To be configured by Application.

- smpCfg_t * pSmpCfg

  Configuration pointer.

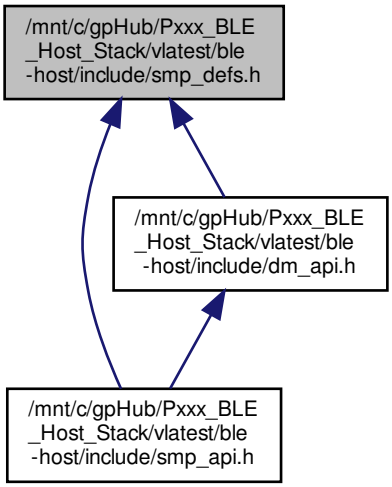
3.1.1 Detailed Description

SMP subsystem API.

3.2 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/ble-host/include/smp_defs.h File Reference

Security manager constants and definitions from the Bluetooth specification.

This graph shows which files directly or indirectly include this file:



Macros

- #define SMP_HDR_LEN 1

  PDU format.
- #define SMP_TIMEOUT 30

  Protocol timeout.
- #define SMP_OOB_LEN 16

  OOB Data length in bytes.
- #define SMP_PIN_LEN 3

  Passkey Pin lenght in bytes.

SMP Encryption Key Size

- #define SMP_KEY_SIZE_MAX 16

  Maximum encryption key size.
- #define SMP_KEY_SIZE_MIN 7

  Minimum encryption key size.

SMP Error Codes

SMP PDU status codes

- #define SMP_ERR_PASSKEY_ENTRY 0x01

User input of passkey failed.

- #define SMP_ERR_OOB 0x02
  OOB data is not available.
- #define SMP_ERR_AUTH_REQ 0x03
  Authentication requirements cannot be met.
- #define SMP_ERR_CONFIRM_VALUE 0x04
  Confirm value does not match.
- #define SMP_ERR_PAIRING_NOT_SUP 0x05
  Pairing is not supported by the device.
- #define SMP_ERR_ENC_KEY_SIZE 0x06
  Insufficient encryption key size.
- #define SMP_ERR_COMMAND_NOT_SUP 0x07
  Command not supported.
- #define SMP_ERR_UNSPECIFIED 0x08
  Unspecified reason.
- #define SMP_ERR_ATTEMPTS 0x09
  Repeated attempts.
- #define SMP_ERR_INVALID_PARAM 0x0A
  Invalid parameter or command length.
- #define SMP_ERR_DH_KEY_CHECK 0x0B
  DH Key check did not match.
- #define SMP_ERR_NUMERIC_COMPARISON 0x0C
  Numeric comparison did not match.
- #define SMP_ERR_BR_EDR_IN_PROGRESS 0x0D
  BR/EDR in progress.
- #define SMP_ERR_CROSS_TRANSPORT 0x0E
  BR/EDR cross transport key generation not allowed.

Proprietary Error Codes

Internal error codes not sent in any SMP PDU.

- #define SMP_ERR_MEMORY 0xE0
  Out of memory.
- #define SMP_ERR_TIMEOUT 0xE1
  Transaction timeout.

SMP PDU Codes

SMP PDU Code describing command received or sent.

- #define SMP_CMD_PAIR_REQ 0x01
  Pairing request.
- #define SMP_CMD_PAIR_RSP 0x02
  Pairing response.
- #define SMP_CMD_PAIR_CNF 0x03
  Pairing confirm.
- #define SMP_CMD_PAIR_RAND 0x04
  Pairing random.
- #define SMP_CMD_PAIR_FAIL 0x05
  Pairing failed.
- #define SMP_CMD_ENC_INFO 0x06
  Encryption information.
- #define SMP_CMD_MASTER_ID 0x07
  Master identification.
- #define SMP_CMD_ID_INFO 0x08
  Identity information.
- #define SMP_CMD_ID_ADDR_INFO 0x09
  Identity address information.

- #define SMP_CMD_SIGN_INFO 0x0A
  Signing information.
- #define SMP_CMD_SECURITY_REQ 0x0B
  Security fequest.
- #define SMP_CMD_PUBLIC_KEY 0x0C
  Public Kkey.
- #define SMP_CMD_DHKEY_CHECK 0x0D
  DH Key check.
- #define SMP_CMD_KEYPRESS 0x0E
  User key press.
- #define SMP_CMD_MAX 0x0F
  Command code maximum.

SMP PDU Packet Lengths

Fixed length of the PDU to be sent.

- #define SMP_PAIR_REQ_LEN 7
  Pairing request message length.
- #define SMP_PAIR_RSP_LEN 7
  Pairing response message length.
- #define SMP_PAIR_CNF_LEN 17
  Pairing confirmation message length.
- #define SMP_PAIR_RAND_LEN 17
  Pairing random message length.
- #define SMP_PAIR_FAIL_LEN 2
  Pairing fail message length.
- #define SMP_ENC_INFO_LEN 17
  Encryption information message length.
- #define SMP_MASTER_ID_LEN 11
  Master identification messagelength.
- #define SMP_ID_INFO_LEN 17
  Identity information message length.
- #define SMP_ID_ADDR_INFO_LEN 8
  Identity address information message length.
- #define SMP_SIGN_INFO_LEN 17
  Signing information message length.
- #define SMP_SECURITY_REQ_LEN 2
  Security request message length.
- #define SMP_PUB_KEY_MSG_LEN (1 + 2∗SMP_PUB_KEY_LEN)
  Public key message length.
- #define SMP_DHKEY_CHECK_MSG_LEN (1 + SMP_DHKEY_CHECK_LEN)
  Diffie-Hellman key check message length.
- #define SMP_KEYPRESS_MSG_LEN 2
  Keypress message length.

SMP I/O Capabilities

I/O capabilities codes to be set for SMP_CMD_PAIR_REQ and SMP_CMD_PAIR_RSP

- #define SMP_IO_DISP_ONLY 0x00
  Display only.
- #define SMP_IO_DISP_YES_NO 0x01
  Display yes/no.
- #define SMP_IO_KEY_ONLY 0x02
  Keyboard only.
- #define SMP_IO_NO_IN_NO_OUT 0x03
  No input, no output.
- #define SMP_IO_KEY_DISP 0x04

Keyboard display.

SMP OOB Data Flag

Out-of-Band codes to be set for SMP_CMD_PAIR_REQ and SMP_CMD_PAIR_RSP

- #define SMP_OOB_DATA_NONE 0x00
  No OOB data from the remote device is present.
- #define SMP_OOB_DATA_PRESENT 0x01
  OOB data from the remote device is present.

SMP Authentication Requirements Flags

Authentication Requirements Flags to be set for SMP_CMD_PAIR_REQ and SMP_CMD_PAIR_RSP.

- #define SMP_AUTH_BOND_MASK 0x03
  Mask for bonding bits.
- #define SMP_AUTH_BOND_FLAG 0x01
  Bonding requested.
- #define SMP_AUTH_MITM_FLAG 0x04
  MITM (authenticated pairing) requested.
- #define SMP_AUTH_SC_FLAG 0x08
  LE Secure Connections requested.
- #define SMP_AUTH_KP_FLAG 0x10
  Keypress notifications requested.

SMP Key Distribution Flags

Flags of security keys this device is requesting to be distribution once pairing completes.

- #define SMP_KEY_DIST_ENC 0x01
  Distribute LTK.
- #define SMP_KEY_DIST_ID 0x02
  Distribute IRK.
- #define SMP_KEY_DIST_SIGN 0x04
  Distribute CSRK.
- #define SMP_KEY_DIST_MASK (SMP_KEY_DIST_ENC | SMP_KEY_DIST_ID | SMP_KEY_DIST_SIGN)
  Key distribution mask.

SMP LE Secure Connection Keypress Codes

Keypress codes found in SMP_CMD_KEYPRESS PDU to be sent on the respective action when the SMP_AUTH_KP_FLAG is set in both the SMP_CMD_PAIR_REQ and SMP_CMD_PAIR_RSP.

- #define SMP_PASSKEY_ENTRY_STARTED 0x00
  Passkey entry started keypress type.
- #define SMP_PASSKEY_DIGIT_ENTERED 0x01
  Passkey digit entered keypress type.
- #define SMP_PASSKEY_DIGIT_ERASED 0x02
  Passkey digit erased keypress type.
- #define SMP_PASSKEY_CLEARED 0x03
  Passkey cleared keypress type.
- #define SMP_PASSKEY_ENTRY_COMPLETED 0x04
  Passkey entry complete keypress type.

SMP Value Length Constants

Lengths of various keys and values.

- #define SMP_RAND_LEN 16
  Random number length.
- #define SMP_CONFIRM_LEN 16
  Confirm number length.
- #define SMP_KEY_LEN 16
  Key length.
- #define SMP_RAND8_LEN 8
  Random 8-byte number length.
- #define SMP_PRIVATE_KEY_LEN 32
  Secure connections private key length.
- #define SMP_PUB_KEY_LEN 32
  Secure connecdtions public key length.
- #define SMP_DHKEY_LEN 32
  Secure connection Diffie-Hellman key length.
- #define SMP_DHKEY_CHECK_LEN 16
  Secure connection Diffie-Hellman key check length.

CMAC Input Lengths Constants

Input lengths of SMP cryptopgraphic toolbox functions.

- #define SMP_F4_TEXT_LEN (SMP_PUB_KEY_LEN ∗ 2 + 1)
  F4 input length.
- #define SMP_G2_TEXT_LEN (SMP_PUB_KEY_LEN ∗ 2 + SMP_RAND_LEN)
  G2 input length.
- #define SMP_F5_TKEY_TEXT_LEN (SMP_DHKEY_LEN)
  F5 Temporary key input length.
- #define SMP_F5_TEXT_LEN (9 + 2∗BDA_ADDR_LEN + 2∗SMP_RAND_LEN)
  F5 input length.
- #define SMP_F6_TEXT_LEN (2∗BDA_ADDR_LEN + 3∗SMP_RAND_LEN + 5)
  F6 input length.

3.2.1 Detailed Description

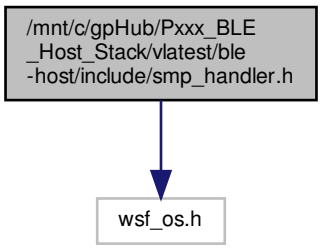
Security manager constants and definitions from the Bluetooth specification.

3.3 /mnt/c/gpHub/Pxxx_BLE_Host_Stack/vlatest/ble-host/include/smp_handler.h File Reference

Interface to SMP event handler.

```
#include "wsf_os.h"
```

Include dependency graph for smp_handler.h:



Functions

SMP Event Handling

Message passing interface to SMP from other tasks through WSF.

- void SmpHandlerInit (wsfHandlerId_t handlerId)
  SMP handler init function called during system initialization.
- void SmpHandler (wsfEventMask_t event, wsfMsgHdr_t *pMsg)
  WSF event handler for SMP.

3.3.1 Detailed Description

Interface to SMP event handler.

Index