



ETI Encryption Supplement

Version 1.0

18th March 2024.

Contents

1	Introduction	3
2	Purpose of the document	3
3	Approach.....	3
4	Process flow	3
5	Proposed Changes <i>in</i> Message Structures	4
5.1	Gateway Response (10022).....	4
5.2	Session Registration Request (10053).....	5
5.3	Session Registration Response (10054).....	6
6	Encryption of Other Request and Response Messages	6
6.1	New Order Single Request.....	7
6.2	Subscription Request:.....	11
6.3	Subscription Response:	12
7	Encryption Summary.....	13
8	Support for TLS and non-TLS version	13
9	TLS 1.3 Implementation using Open SSL.....	14
9.1	Function Calls for TLS v1.3 Implementation at Connection Gateway	14
9.1.1	Initialization of SSL/TLS Library:	14
9.1.2	Initiate SSL/TLS connection on top of the TCP socket:	14
9.1.3	Validate the authenticity of Gateway Certificate.....	15
9.2	Function Calls for Symmetric Encryption/Decryption at Gateway.....	15
9.2.1	Encryption	15
9.2.2	Decryption:.....	16
10	Document Control Page.....	16

1 Introduction

The Interactive communication between member trading application and exchange is supported through ETI protocol which is binary format fixed data structures. The data exchange between member trading application and exchange happens via a dedicated leased line and is a secure connection. However, since the data structures transmitted on wire are unencrypted and thus there may exist a remote possibility of data security getting compromised in transit. To plug this remote possibility, it is proposed to have the data structures encrypted and sent over the wire so that security of the data in transit is maintained.

2 Purpose of the document

The purpose of this document is to provide guidance for programmers already familiar with ETI protocol and have fair know how of the ETI framework of communication. For beginners this document needs to be read in conjunction with ETI API version 1.6.8 for having better clarity on the message flow mentioned in this document.

3 Approach

It is proposed to implement encryption at exchange using TLS 1.3. A 32-byte secret key and a 16-byte initialization vector (IV) will be provided in connection response on a TLS channel. Once the key and IV is received then the member trading application can continue to communicate with exchange over normal TCP/IP channel wherein all the messages sent by member application to exchange needs to be encrypted using the key and IV. Similarly, all the responses or unsolicited messages sent by exchange will be encrypted with the same key and IV. The decryption process should happen at the receiver end using the same key and IV.

4 Process flow

1. Exchange will generate a self-signed certificate, which will be provided to members via extranet. This CA certificate will be same for all members and will be changing on a periodic basis.
2. At the time of TLS Handshaking, authenticity of received certificate via socket will be validated with the help of certificate given to the members via extranet.
3. Gateway request will be sent by the member at connection gateway and in response of that member will receive a 32-byte secret key and a 16-byte initialization vector. There is no change in **Connection Gateway Request (10020)**. The **Connection Gateway Response (10022)** will have 2 additional fields. i.e., Security Key and Initialisation Vector.
4. The member application will terminate the TLS connection with the connection gateway initiate a TCP socket with the gateway whose IP address and port values are received in the **Connection Gateway Response (10022)**.
5. On the established TCP connection, the member application should send the **Session Registration Request (10053)** in unencrypted form. This should be the first message on the socket and the field *MsgSeqNum (34)* should be set to 1. This will register the session for the encrypted communication between the exchange and member application. The exchange

will respond with the **Session Registration Response (10054)** with either success or failure along with the reason of failure.

6. All the subsequent messages exchanged between exchange and member application should be encrypted by the sender and should be decrypted by the receiver using the key and IV which was exchanged in **Connection Gateway Response (10022)**.

5 Proposed Changes *in* Message Structures

5.1 Gateway Response (10022)

This message confirms the Connection Gateway request. Two additional fields i.e., *SecurityKey (820)* and *InitializationVector (821)* are added in the message.

Tag	Field Name	Data Type	Size	Description
<MessageHeaderOut>				
9	BodyLen	unsigned int	4	Number of bytes for the message, including this field
28500	TemplateID	unsigned int	2	Unique identifier for a BSE ETI message layout. Value: 10022 (ApplicationMessageRequestAck, MsgType = BX)
25017	Pad2	Fixed String	2	not used
<ResponseHeader>				
5979	RequestTime	UTCTimestamp	8	Gateway request in timestamp.
52	SendingTime	UTCTimestamp	8	Gateway response out timestamp
34	MsgSeqNum	unsigned int	4	Message sequence number used by the participant for requests sent to the gateway
25019	Pad4	Fixed String	4	not used
<Message Body>				
28644	GatewayID	unsigned int	4	IP address of the Primary Application Gateway where the member application can establish an active BSE ETI session. Note: Binary values are presented in little endian byte order.
28645	GatewaySubID	unsigned int	4	Port number to be used for the assigned Primary Application Gateway. Note: Binary values are presented in little endian byte order
28725	SecondaryGatewayID	unsigned int	4	IP address of the Secondary Application Gateway where the member application can establish an active BSE ETI session. Note: Binary values are presented in little endian byte order
28726	SecondaryGatewaySub	unsigned int	4	Port number to be used for the assigned Secondary Application Gateway. Note:

				Binary values are presented in little endian byte order.
28730	SessionMode	unsigned int	1	
339	TradSesMode	unsigned int	1	
820	SecurityKey	String	32	To be used while encrypting/decrypting the data at gateway
821	InitializationVector	String	16	To be used while encrypting/decrypting the data at gateway
25021	Pad6	Fixed String	6	not used

5.2 Session Registration Request (10053)

This is a new request and needs to be sent to the gateway after establishing the TCP socket with the gateway whose IP address and port values are received in the Connection Gateway Response (10022).

Tag	Field Name	Data Type	Size	Description
<MessageHeaderIn>				
9	BodyLen	unsigned int	4	Number of bytes for the message, including this field
28500	TemplateID	unsigned int	2	Unique identifier for a BSE ETI message layout. Value: 10053
25028	NetworkMsgId	Fixed String	8	not used
25017	Pad2	Fixed String	2	not used
<RequestHeader>				
34	MsgSeqNum	unsigned int	4	Message sequence number used by the participant for requests sent to the gateway.
50	SenderSubID	unsigned int	4	User Id
<Message Body>				
200055	PartyIDSessionID	unsigned int	4	Session ID.
25019	Pad4	Fixed String	4	Not used
30646	Filler1	unsigned int	8	Not used

5.3 Session Registration Response (10054)

This message confirms the session registration request.

Tag	Field Name	Data Type	Size	Description
<MessageHeaderOut>				
9	BodyLen	unsigned int	4	Number of bytes for the message, including this field
28500	TemplateID	unsigned int	2	Unique identifier for a BSE ETI message layout. Value:10054
25017	Pad2	Fixed String	2	Not used
<Response Header>				
5979	RequestTime	UTCTimestamp	8	Gateway request in timestamp
52	SendingTime	UTCTimestamp	8	Gateway response out timestamp
34	MsgSeqNum	unsigned int	4	Message sequence number used by the participant for requests sent to the gateway
25019	Pad4	Fixed String	4	Not used
<Message Body>				
823	Status	Flag	1	Status of request
30354	VarTextLen	Counter	2	Number of bytes for field VarText (30355).
30355	VarText	Variable String	2000	Error text. Valid characters: \x09,\x0A,\x0D, \x20,\x22-\x7B,\x7D,\x7E

6 Encryption of Other Request and Response Messages

The encryption is required for all the messages except the Connection gateway request (as connection would be TLS) and Session Registration request. The encryption is done on the message body for all the messages. The message header should be left unencrypted. Different messages may have different header sizes. Following are 2 sample messages illustrated below to explain the encryption.

6.1 New Order Single Request

Tag	Field Name	Req'd	Len	Ofs	Data Type	Description	
<i><MessageHeaderIn></i>							
9	BodyLen	Y	4	0	unsigned Int	Number of bytes for the message, including this field.	Should not be encrypted.
28500	TemplateID	Y	2	4	unsigned Int	Unique Identifier for a BSE ETI message layout. Value: 10100 (NewOrderSingle, MsgType = D)	
25028	NetworkMsgID	U	8	6	Fixed String	not used	
25017	Pad2	U	2	14	Fixed String	not used	
<i><RequestHeader></i>							
34	MsgSeqNum	Y	4	16	unsigned Int	Message sequence number used by the participant for requests sent to the gateway.	Should be encrypted.
50	SenderSubID	Y	4	20	unsigned Int	User ID.	
<i><Message Body></i>							
44	Price	N	8	24	PriceType	Limit price. Required if OrdType (40) is Limit (2) or Stop Limit (4).	Should be encrypted
99	StopPx	N	8	32	PriceType	Stop price. Required if OrdType (40) is Stop (3) or Stop (4).	
28740	MaxPricePercentage	N	8	40	PriceType	Indicates the protection range for market order. Required if OrderType (40) is Market(5).	

142	SenderLocationID	Y	8	48	unsigned Int	The location ID of the order from where it is originated						
11	ClOrdID	N	8	56	unsigned Int	Client Order ID: Unique participant defined order request identifier; used for client order id chaining.						
30646	Filler1	N	8	64	unsigned Int	not used						
30647	Filler2	N	4	72	unsigned Int	not used						
526	MessageTag	Y	4	76	signed Int	User defined free field. The field is echoed in the trade notifications, listener data, session data.						
38	OrderQty	Y	4	80	Qty	Total Order Quantity.						
210	MaxShow	N	4	84	Qty	The quantity to be made visible in the market data.						
432	ExpireDate	N	4	88	LocalMktDate	not used						
1300	MarketSegmentID	N	4	92	signed Int	Product Identifier.						
30048	SimpleSecurityID	Y	4	96	unsigned Int	Instrument Identifier for simple instruments. Should be filled with the 4 least significant bytes of SecurityID (48) provided in BSE reference data.						
25029	RegulatoryID	N	4	100	unsigned Int	not used.						
30652	Filler4	N	2	104	unsigned Int	not used						
20096	PartyIDTakeUpTrading-Firm	N	5	106	Fixed String	not used. Valid characters: A-Z,0-9,\x20						
20013	PartyIDOrder-OriginationFirm	N	7	111	Fixed String	not used. Valid characters: A-Z,0-9,\x20						
20032	PartyIDBeneficiary	N	9	118	Fixed String	not used. Valid characters: A-Z,0-9,\x20						
581	AccountType	Y	1	127	unsigned Int	<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>20</td> <td>Own</td> </tr> <tr> <td>30</td> <td>Client</td> </tr> </tbody> </table>	Value	Description	20	Own	30	Client
Value	Description											
20	Own											
30	Client											
28703	AppSeqIndicator	Y	1	128	unsigned Int	<p>Indicates if the order is a Lean Order. Application sequencing is limited for Lean Orders.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Lean Order</td> </tr> <tr> <td>1</td> <td>Standard (non lean) Order</td> </tr> </tbody> </table>	Value	Description	0	Lean Order	1	Standard (non lean) Order
Value	Description											
0	Lean Order											
1	Standard (non lean) Order											

54	Side	Y	1	129	unsigned Int	Side of the order. <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Buy</td> </tr> <tr> <td>2</td> <td>Sell</td> </tr> <tr> <td>3</td> <td>Recall</td> </tr> <tr> <td>4</td> <td>Early Return</td> </tr> </tbody> </table>	Value	Description	1	Buy	2	Sell	3	Recall	4	Early Return		
Value	Description																	
1	Buy																	
2	Sell																	
3	Recall																	
4	Early Return																	
40	OrdType	Y	1	130	unsigned Int	Order type. <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>2</td> <td>Limit</td> </tr> <tr> <td>3</td> <td>Stop Market</td> </tr> <tr> <td>4</td> <td>Stop Limit</td> </tr> <tr> <td>5</td> <td>Market</td> </tr> <tr> <td>6</td> <td>Block Deal</td> </tr> </tbody> </table>	Value	Description	2	Limit	3	Stop Market	4	Stop Limit	5	Market	6	Block Deal
Value	Description																	
2	Limit																	
3	Stop Market																	
4	Stop Limit																	
5	Market																	
6	Block Deal																	
28710	PriceValidityCheckType	Y	1	131	unsigned Int	not used. <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>None</td> </tr> </tbody> </table>	Value	Description	0	None								
Value	Description																	
0	None																	
59	TimeInForce	Y	1	132	unsigned Int	Execution and trading restriction parameters supported by BSE. <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Day (GFD)</td> </tr> <tr> <td>3</td> <td>Immediate or Cancel (IOC)</td> </tr> <tr> <td>7</td> <td>Session (GFS)</td> </tr> </tbody> </table>	Value	Description	0	Day (GFD)	3	Immediate or Cancel (IOC)	7	Session (GFS)				
Value	Description																	
0	Day (GFD)																	
3	Immediate or Cancel (IOC)																	
7	Session (GFS)																	
18	ExecInst	Y	1	133	unsigned Int	Instructions for order handling. <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Persistent Order (FIX value "H")</td> </tr> <tr> <td>2</td> <td>Non-persistent Order (FIX value "Q")</td> </tr> <tr> <td>5</td> <td>Persistent BOC Order (FIX value "H 6")</td> </tr> <tr> <td>6</td> <td>Non-persistent BOC Order (FIX value "Q 6")</td> </tr> </tbody> </table>	Value	Description	1	Persistent Order (FIX value "H")	2	Non-persistent Order (FIX value "Q")	5	Persistent BOC Order (FIX value "H 6")	6	Non-persistent BOC Order (FIX value "Q 6")		
Value	Description																	
1	Persistent Order (FIX value "H")																	
2	Non-persistent Order (FIX value "Q")																	
5	Persistent BOC Order (FIX value "H 6")																	
6	Non-persistent BOC Order (FIX value "Q 6")																	
30651	STPCFlag	Y	1	134	unsigned Int	Indicates the preference of order to be cancelled if self-trade is encountered. Preference not applicable in a modification requests. <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Passive</td> </tr> <tr> <td>1</td> <td>Active</td> </tr> </tbody> </table>	Value	Description	0	Passive	1	Active						
Value	Description																	
0	Passive																	
1	Active																	

30653	RolloverFlag	N	1	135	unsigned Int	not used
625	TradingSessionSubID	N	1	136	unsigned Int	not used
1815	TradingCapacity	Y	1	137	unsigned Int	not used. Must be sent as 1
1	Account	Y	2	138	Fixed String	not used. Must be sent as A1 Valid characters: 1-9, \x41, \x47, \x4D, \x50
77	PositionEffect	Y	1	140	char	must be set to C Valid characters: \x01-\x7E
20075	PartyIDLocationID	N	2	141	Fixed String	not used. Valid characters: \x01-\x7E
1031	CustOrderHandlingInst	N	1	143	Fixed String	not used. Valid characters: \x20, \x22-\x7B, \x7D, \x7E
25015	RegulatoryText	N	20	144	Fixed String	This field is used to provide additional regulatory information (according to respective rules & regs, circulars and/or bilateral coordination between participant and Trading Surveillance Office). Valid characters: \x20, \x22-\x7B, \x7D, \x7E
30649	AlgoID	N	16	164	Fixed String (0-terminable)	Unique identifier of the algorithm. Valid characters: A-Z, 0-9
58	ClientCode	Y	12	180	Fixed String (0-terminable)	The Unique Client Code (UCC) of the person for whom the order is entered. For Proprietary Account Client code should be OWN always. Valid characters: A-Z, 0-9, \x2A, \x2D, \x2E, \x2F, \x5C
30642	CPCode	N	12	192	CharText	The Participant code. Valid characters: \x00, \x21, \x23, \x24, \x28, \x29, \x2A, \x2B, \x2D, \x2E, \x2F, \x30-\x39, \x3A, \x3B, \x3D, \x3E, \x3F, \x40, \x41-\x5A, \x5C, \x5D, \x5F, \x60, \x7B, \x7D
25009	FreeText3	N	12	204	CharText	Free-format text field for trader-specific or customer-related comments. Valid characters: \x00, \x21, \x23, \x24, \x28, \x29, \x2A, \x2B, \x2D, \x2E, \x2F, \x30-\x39, \x3A, \x3B, \x3D, \x3E, \x3F, \x40, \x41-\x5A, \x5C, \x5D, \x5F, \x60, \x7B, \x7D

6.2 Subscription Request:

Tag	Field Name	Req'd	Len	Ofs	Data Type	Description													
<MessageHeaderIn>																			
9	BodyLen	Y	4	0	unsigned Int	Number of bytes for the message, including this field.	Should not be encrypted												
28500	TemplatID	Y	2	4	unsigned Int	Unique identifier for a BSE ETI message layout. Value: 10025 (ApplicationMessageRequest, MsgType = BW)													
25028	NetworkMsgID	U	8	6	Fixed String	not used													
25017	Pad2	U	2	14	Fixed String	not used													
<RequestHeader>																			
34	MsgSeqNum	Y	4	16	unsigned Int	Message sequence number used by the participant for requests sent to the gateway.	Should be encrypted.												
50	SenderSubID	U	4	20	unsigned Int	not used													
<Message Body>																			
25001	SubscriptionScope	N	4	24	unsigned Int	- Subscription Scope : Value - Trade : Session ID - Trade (Business Unit) : No Value - News broadcast : 1 - Risk Control broadcast: No Value - Service Availability : Partition ID or No Value	Should be encrypted												
1355	RefApplID	Y	1	28	unsigned Int	Application identifier of a BSE ETI data stream. <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Trade</td> </tr> <tr> <td>2</td> <td>News</td> </tr> <tr> <td>3</td> <td>Service Availability</td> </tr> <tr> <td>5</td> <td>Listener Data</td> </tr> <tr> <td>0</td> <td>Trade Enhancement</td> </tr> </tbody> </table>		Value	Description	1	Trade	2	News	3	Service Availability	5	Listener Data	0	Trade Enhancement
Value	Description																		
1	Trade																		
2	News																		
3	Service Availability																		
5	Listener Data																		
0	Trade Enhancement																		
25018	Pad3	U	3	29	Fixed String	not used													

6.3 Subscription Response:

Subscription Response							
Tag	Field Name	Req'd	Len	Ofs	Data Type	Description	Remarks
<MessageHeaderOut>							
9	BodyLen	Y	4	0	unsigned int	Number of bytes for the message, including this field.	Should not be decrypted
28500	TemplateID	Y	2	4	unsigned int	Unique identifier for a BSE ETI message layout. Value: 10005 (ApplicationMessageRequestAck, MsgType =BX)	
25017	Pad2	U	2	6	Fixed String	not used	
<ResponseHeader>							
5979	RequestTime	Y	8	8	UTCTimestamp	Gateway request in timestamp.	Should be decrypted
52	SendingTime	Y	8	16	UTCTimestamp	Gateway response in timestamp.	
34	MsgSeqNum	Y	4	24	unsigned int	Message sequence number used by the participant for requests sent to the gateway.	
25019	Pad4	U	4	28	Fixed String	not used	
<Message Body>							
28727	ApplSubID	Y	4	32	unsigned int	Unique ID assigned by the BSE system during broadcast subscription in order to link broadcasts to the related subscription	Should be decrypted
25019	Pad4	U	4	36	Fixed String	not used	

7 Encryption Summary

Sr. No.	Message	Fields Encrypted	Mode	Key Size (bytes)	IV size (bytes)	Key and IV
1.	Connection Gateway Request (10020)	Unencrypted SSL WRITE	TLS v1.3	Protocol specific	Protocol specific	Decided during TLS Handshake
2.	Connection Gateway Response (10022)	Unencrypted SSL READ	TLS v1.3	Protocol specific	Protocol specific	Decided during TLS Handshake
3.	Session Registration Request (10053)	Unencrypted	NA	-	-	-
4.	Session Registration Response (10054)	Unencrypted	NA	-	-	-
5.	All Request messages except Session Registration Request	Entire Msg body Encrypted except for headers (which is of 16 bytes)	AES_256_GCM	32	16	Use Key and IV received in Gateway Response
6.	All Response messages except Session Registration Response	Entire Msg body Encrypted except for headers (which is of 8 bytes)	AES_256_GCM	32	16	Use Key and IV received in Gateway Response

8 Support for TLS and non-TLS version

The exchange will deploy the TLS related changes in simulation and production, and it will support the existing version (without encryption) and TLS version (with encryption) simultaneously in production for few weeks. During this period the member applications which are ready with the TLS related changes need not wait till last date to deploy their changes in production. The field

DefaultCstmAppVerID (1408) in connection gateway request determines the version the member application is willing to interact with.

The member application ready with TLS changes mentioned in this document needs to set the value of the field *DefaultCstmAppVerID(1408)* to 2.4 in the connection gateway request. If the value is set to 2.4 then the trading system will expect the next request as Session registration request and all other messages needs to be encrypted as per the logic defined in this document.

9 TLS 1.3 Implementation using Open SSL.

The implementation of Encryption and decryption require the Open SSL and TLS 1.3 installed.

The open-source download link for OpenSSL 3.0 for Linux platform is as follow. This download contains the source code, user must configure and build it to generate libraries.

<https://www.openssl.org/source/openssl-3.0.12.tar.gz>

The following are illustrative prototypes for implementation of TLS using OpenSSL (ssl) library calls.

- It is recommended to use OpenSSL's TLS implementation.
- The minimum supported version for TLS protocol should be set to 1.3.

9.1 Function Calls for TLS v1.3 Implementation at Connection Gateway

9.1.1 Initialization of SSL/TLS Library:

- ***SSL_library_init ()*** - Initializes the SSL/TLS library and registers the available SSL/TLS ciphers.
- ***OpenSSL_add_all_algorithms ()*** - It loads algorithms and initializes the data structures.
- ***SSL_load_error_strings ()*** – Initializes and loads error strings for OpenSSL library errors.
- ***SSL_CTX_new (TLS_client_method ())*** - Creates a new SSL_CTX object, which holds SSL configuration data.
- ***SSL_CTX_set_min_proto_version (SSL_CTX *ctx, int version)*** - Sets the minimum acceptable protocol version.
- ***SSL_CTX_set_max_proto_version (SSL_CTX *ctx, int version)*** - Sets the maximum acceptable protocol version.

9.1.2 Initiate SSL/TLS connection on top of the TCP socket:

- ***SSL_new (SSL_CTX *ctx)*** - It creates a new SSL structure, representing a connection using the TLS protocol, providing the foundation for secure communication over a network.
- ***SSL_set_fd (SSL *ssl, int fd)*** - It associates a file descriptor (socket) with an SSL object, enabling TLS communication over the specified file descriptor.
- ***SSL_connect (SSL *ssl)*** - It initiates TLS handshake process to establish a secure connection with the peer.

9.1.3 Validate the authenticity of Gateway Certificate

Member can use any of the following approaches to verify the authenticity of the gateway certificate.

1. Approach - 1

- `X509 *cert = SSL_get_peer_certificate (ssl)` - Used to retrieve the peer certificate during the TLS handshake.
- `SSL_get_verify_result (*ssl)` - Used to retrieve the result of the peer certificate verification performed during the TLS handshake.

2. Approach - 2

- `X509_STORE_new ()` - This function returns a new X509_STORE.
- `X509_STORE_CTX_new ()` - This function returns a newly initialised X509_STORE_CTX.
- `X509_STORE_load_locations (X509_STORE *ctx, const char *file, const char *dir)` - Configure files and directories used by a certificate store. The path to the CA certificate (gw_ca_cert.pem) should be specified. The CA certificate (gw_ca_cert.pem) will be provided by the Exchange for validation of Gateway certificate.
- `X509_STORE_CTX_init (X509_STORE_CTX *ctx, X509_STORE *trust_store, X509 *target, STACK_OF(X509) *untrusted)` - This function returns a newly initialised X509_STORE_CTX structure.
- `X509_verify_cert (X509_STORE_CTX *ctx)` - This function builds and verifies the X509 certificate chain.

Send and Receive messages from the TLS Socket

- `SSL_write (SSL *ssl, const void *buf, int num)` - It is used to send data securely over an established TLS connection.
- `SSL_read (SSL *ssl, void *buf, int num)` - It is used to securely receive data from the peer over the same TLS connection.

9.2 Function Calls for Symmetric Encryption/Decryption at Gateway

The following prototypes implements AES 256 GCM encryption algorithm using OpenSSL (crypto) library calls.

9.2.1 Encryption

a. Initialization

- `EVP_CIPHER_CTX **ctx` - Used to reference the context structure for symmetric encryption and decryption operations in OpenSSL.
- `EVP_CIPHER_CTX_new ()` - Used to create a new EVP_CIPHER_CTX structure, providing a context for symmetric encryption and decryption.

- iii. `EVP_EncryptInit_ex (*ctx, EVP_aes_256_gcm (), NULL, NULL, NULL)` - Used to initialize a symmetric encryption operation with AES 256 GCM encryption algorithm.
- iv. `EVP_CIPHER_CTX_ctrl (*ctx, EVP_CTRL_GCM_SET_IVLEN, iv_length, NULL)` – Set the IV length to 16 bytes.
- v. `EVP_EncryptInit_ex (*ctx, NULL, NULL, key, iv)` - Used to initialize a symmetric encryption operation with additional parameters, such as key, IV in an existing `EVP_CIPHER_CTX` context structure.
- vi. Do not encrypt first 16 bytes 'MessageHeaderIn' while encrypting.

b. Encryption

- i. Int len = length of the output plaintext.
- ii. `EVP_EncryptUpdate (ctx, ciphertext, &len, plaintext, plaintext_len)` - Perform symmetric encryption on input data in chunks.

9.2.2 Decryption:

a. Initialization

- i. `EVP_CIPHER_CTX **ctx` - Used to reference the context structure for symmetric encryption and decryption operations in openssl.
- ii. `EVP_DecryptInit_ex (*ctx, EVP_aes_256_gcm (), NULL, NULL, NULL)` - Used to initialize a symmetric decryption operation, setting up the encryption algorithm with AES 256 GCM.
- iii. `EVP_CIPHER_CTX_ctrl (*ctx, EVP_CTRL_GCM_SET_IVLEN, iv_length, NULL)` – Set the IV length to 16 bytes.
- iv. `EVP_DecryptInit_ex (*ctx, NULL, NULL, key, iv)` – Used to initialize a symmetric encryption operation with additional parameters, such as key, IV in an existing `EVP_CIPHER_CTX` context structure.
- v. Do not decrypt first 8 bytes 'MessageHeaderOut' while decrypting.

b. Decryption

- i. `EVP_DecryptUpdate (ctx, plaintext, &len, ciphertext, ciphertext_len)` - Used to perform symmetric decryption on input data.

10 Document Control Page

Sr. No.	Version	Date	Description
1.	1.0	15 th March 2024	Original Document