CHAPTER 11

Security Manager (SM)

11.1 Introduction

The security manager defines the procedures for pairing, authentication, and encryption between LE devices. This is needed once a link layer connection has been established and if security is requested on that particular connection. As shown in Figure 11.1, the security manager is located above L2CAP in the LE architecture. It uses the services of L2CAP to carry out its different procedures.

The security manager protocol is used to generate and store various keys (like encryption and identity). It uses a key distribution approach where each device generates and controls the keys it distributes. The security manager is also responsible for generating random addresses and resolving random addresses to known device identities. The security manager allows for keys from 56-bits to 128-bit length in 8-bit steps. The key length is defined by the profile or application that requests security. If devices have lower processing power or need less security, then they need not generate all bits of the 128-bit keys. They can generate a lesser number of bits (subject to a minimum of 56-bits) and set the remaining bits to 0.

Specifications 4.2 introduced a major enhancement in security by defining LE secure connections. The method defined by specifications 4.0 are now referred to as LE legacy pairing.

11.2 Security in Host Instead of Controller

One major difference between the security architecture of BR/EDR and LE is that the security in BR/EDR is handled in the link manager, while in the case of LE, the security related procedures are moved to the host. So all procedures related to key generation and distribution of keys are performed by the host. This helps to keep the cost of LE-only controllers low and also provides more flexibility to the host. If the key generation algorithms need to be upgraded (for example, to increase the level of security offered by the device) then only the software in the host needs to be changed without any modification in the controller. The encryption of data just before transmitting the packets over the air is still done by the controller in both LE and BR/EDR.

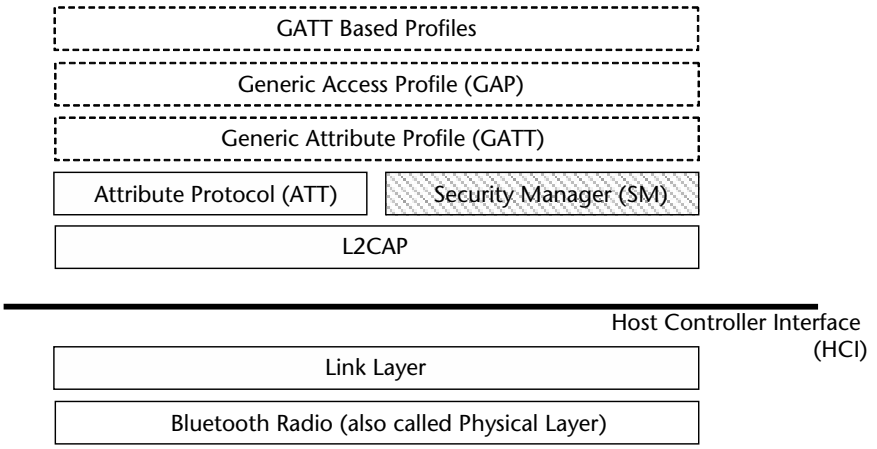

Figure 11.1 Security manager in LE protocol stack.

11.3 Asymmetrical Architecture

The security manager has an asymmetrical architecture. This means that the architecture is designed so that the memory and processing requirements for the responding device are much lower than the memory and resource requirements of the initiating device.

This is another enhancement done by LE to optimize the power consumption of the peripheral devices. In general the initiating device could be a dual mode device like a mobile phone which has much higher memory, processing power, and battery power available. In comparison a peripheral device like a key fob could have limited memory, processing power, and battery power. So the architecture is designed in such a manner that more processing power and memory is used on the mobile phone resulting in lesser memory and processing power used at the key fob end.

11.4 Security Breaches

Before going into the details of security manager, it is important to understand the possible security breaches in a wireless system. These will be explained in this section. Later sections will explain how the security manager helps to prevent these security breaches.

11.4.1 Passive Eavesdropping

Passive Eavesdropping attacks occurs when an attacker starts listening to the data being exchanged between two devices. If the attacker is present at the time of initial pairing, it can listen to the keys that are being exchanged during pairing. After that it can use the same keys to decrypt all the information being sent between the two devices.

Passive eavesdropping is difficult to detect. This is because, the passive eavesdropper can be anywhere in the Bluetooth range (anywhere between 10 meters to

100 meters). So the eavesdropper need not be visible to the user. Eavesdropping can be done, for example, from another room. This is shown in Figure 11.2.

11.4.2 Man-in-the-Middle (MITM) (Active Eavesdropping)

MITM was explained in Chapter 3. The explanation is repeated here for convenience. An MITM attack occurs when a rouge device attacks by sitting in the middle of two devices that want to connect and relays messages between them. The two devices believe that they are directly talking to each other without knowing that all their messages are being intercepted and relayed by a third device which is between them. This is also known as active eavesdropping.

Let's say devices A and B want to make a connection and M is an attacking device (as shown in Figure 11.3). M receives all information from A and relays it to B and vice versa. So, A and B have an illusion that they are directly connected. They are not aware of the existence of M between them. Since M is relaying the

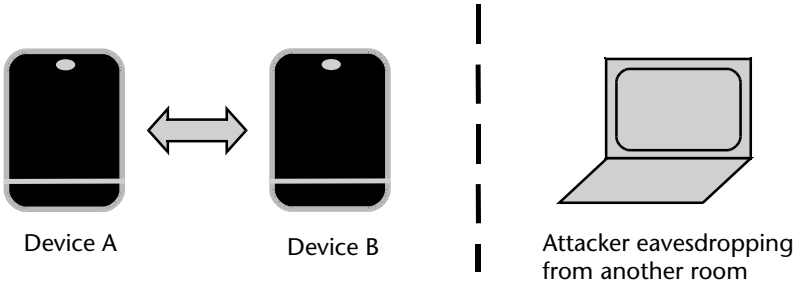

Figure 11.2 Example of Passive Eavesdropping.

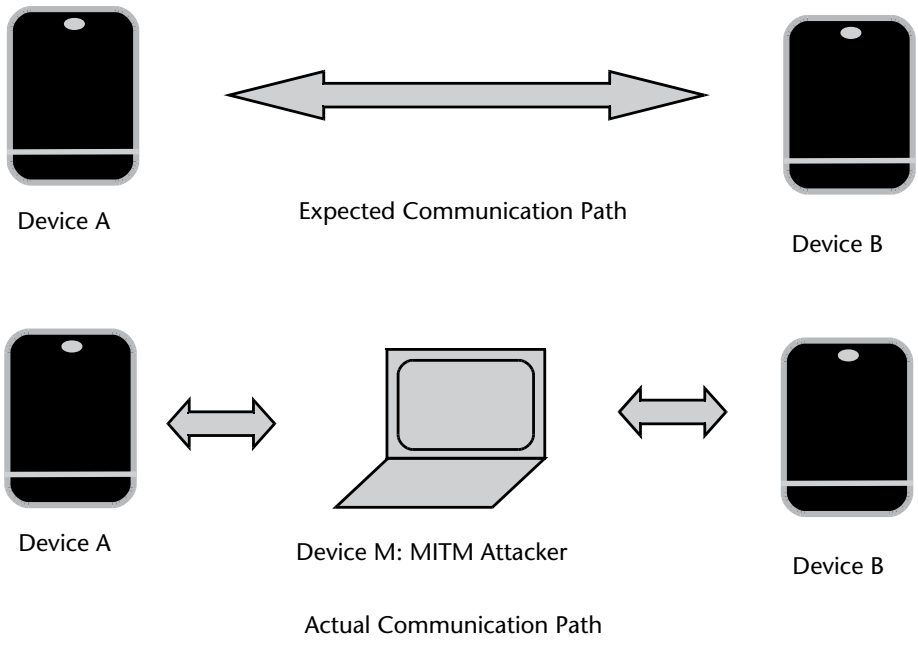

Figure 11.3 Example of MITM attack.

information between the two devices, it can interpret all this information and misuse it. Besides this, M can also attack by inducing rogue information between the two devices.

11.4.3 Tracking

Since many of the LE devices are intended to be carried by the user, an attacker can potentially try to track the transmissions coming from these devices to track the person. For example, if a person is wearing an LE watch or carrying a key fob which are advertising, then an attacker can read those advertising packets and follow the person by just following the advertising packets.

LE provides a privacy feature in which the LE devices can use a random address and change this random address frequently. So only devices that are previously authenticated by the person can resolve that address. This means that only the authenticated devices can map the random address to find out the exact device from which the packets are being transmitted. This prevents tracking from any unauthorized person.

11.5 Pairing Methods

The pairing method used to pair the devices depends on the Input/Output capabilities of the two devices. For example, if one of the devices is capable of entering a 6-digit number and another is capable of displaying a 6-digit number, then the Passkey entry method can be used.

Security Manager provides four types of pairing methods:

1. Just Works.
2. Numeric Comparison (introduced in specifications 4.2 for LE secure connections).
3. Passkey Entry.
4. Out of Band.

11.5.1 Just Works

In the case of Just Works pairing no passkeys are exchanged at the UI level. This is used when at least one of the devices does not have a mechanism to either display a 6-digit number or enter 6-digit numbers. An example of this could be a mono headset which does not have display capability or the ability to enter a passkey.

11.5.2 Numeric Comparison

In this method, the two devices each compute 6-digit confirmation values that are displayed to the users on their respective devices. The users are expected to check that these 6-digit values match and to confirm whether or not there is a match. If there is no match, the protocol aborts.

11.5.3 Passkey Entry

This method is designed for use when one device has input capability and the other device has display capability. In this method, a 6-digit numeric passkey is displayed on one of the devices and the user is asked to enter that passkey on the second device. For example, if a keyboard is being paired to a PC, then the PC may display a 6-digit numeric passkey which can be entered from the keyboard. Another example could be a remote being paired to a TV. The TV can display the passkey which is entered from the remote control.

11.5.4 Out of Band

This is designed for scenarios where an Out-of-Band mechanism can be used to transfer security information. For example, if NFC is used, then the user may touch the two devices to exchange the security information.

These four pairing methods will be explained in further detail later in this chapter.

11.6 Security Properties

Security Manager provides four types of security:

1. LE secure connections pairing.
2. Authenticated MITM protection.
3. Unauthenticated No MITM protection.
4. No security.

The LE secure connections pairing is one of the key enhancements introduced by specifications 4.2. It uses P-256 elliptic curve cryptography and Diffie-Hellman public key exchanges.

The methods of pairing specified by specifications 4.0 and 4.1 are now referred to as LE legacy pairing.

In LE legacy pairing, none of the pairing methods provide protection against passive eavesdropper if the eavesdropper starts listening during the pairing process. This is because, at present, predictable or easily established values are used as temporary keys in the beginning of the pairing. If the pairing is done when the eavesdropper is not listening, then the communication channel is secure from any passive eavesdroppers and further communication can go on without any risk.

11.6.1 LE Secure Connections Pairing

LE secure connections pairing uses P-256 elliptic curve cryptography. This method generates the LTK, which is a 128-bit key that will be used to encrypt the connection after pairing and also for encrypting subsequent connections. These will be explained in detail in Section 11.8.2.5.

11.6.2 Authenticated MITM Protection

In LE legacy pairing, authenticated MITM protection can be achieved by using the passkey entry pairing method or by using Out Of Band (OOB) pairing.

- In the case of the passkey entry pairing method the passkey is displayed on one device and entered from another device. So there is no way that an MITM can relay this information between these two devices.
- In the case of Out Of Band, some other technique apart from Bluetooth is used for exchanging the pairing information. For example, NFC could be used where the two devices touch each other to exchange the pairing information. Here again, if the OOB method being used provides MITM protection, only then the MITM attacks can be prevented.

In LE secure connections pairing method, MITM protection is obtained by numeric comparison method in addition to the two methods listed above.

- *Numeric Comparison method:* In this method, the two devices each compute 6-digit confirmation values that are displayed to the users on their respective devices. The users are expected to check that these 6-digit values match and to confirm whether or not there is a match. If there is no match, the protocol aborts.

11.6.3 Unauthenticated no MITM Protection

Unauthenticated no MITM protection does not provide protection from MITM attacks. This is achieved by using the Just Works pairing method. In this method, no passkeys are exchanged at the UI level. It is still more secure than No security since the keys are exchanged between the two devices though they are not exchanged at the UI level. In this case the keys are automatically generated by the two devices.

11.6.4 No Security

In the no security mode, there is no support for authentication or encryption. In this case the information transferred may not be sensitive. For example, an LE thermometer sending temperature data may use No Security Mode. Similarly No Security Mode may be used to fetch information like the device manufacturer's name, device model number, etc.

11.7 Cryptographic Functions

Security manager provides the following cryptographic functions which serve as enablers for various security operations described later.

1. Security function *e*.
2. Random address function *ah*.
3. Confirm value generation function *c1*.
4. Key generation *s1*.

The following cryptographic operations are defined by specifications 4.2 to support LE Secure Connections:

- Security function AES-CMAC;
- LE Secure Connections Confirm Value Generation function f4;
- LE Secure Connections Key Generation function f5;
- LE Secure Connections Check Value Generation function f6;
- LE Secure Connections Numeric Comparison Value Generation function g2;
- LE Secure Connections Link Key Conversion function h6.

The building block for cryptographic functions ah, c1, and s1 is the security function e.

The building block for the cryptographic functions f4, f5, f6, g2, and h6 is the security function AES-CMAC.

11.7.1 Security Function e

The security function *e* is used to generate 128-bit encrypted data from 128-bit plain text data. It uses the AES-128 bit cypher.

encryptedData = e(key, plaintextData)

This function can be implemented in the host or the controller. If this functionality is implemented in the controller, then the host can use it via the HCI command HCI_LE_Encrypt. This function is used by the next three cryptographic functions.

11.7.2 Random Address Function ah

The random address function *ah* is used in generating a hash value used in the resolvable private address.

11.7.3 Confirm Value Generation Function c1

The confirm value generation function *c1* is used to generate the confirm values used during the confirm procedure of the Security Manager Protocol.

11.7.4 Key Generation Function s1

The key generation function *s1* is used to generate the Short Term Key (STK) during the pairing process.

11.7.5 Security Function AES-CMAC

Cipher-based message authentication code (CMAC) is a block cipher-based message authentication code algorithm. It provides a stronger assurance of data integrity than a checksum or an error-detecting code. While checksum and error-detecting

code only detect accidental modifications to data, CMAC is designed to detect intentional, unauthorized modifications of data, as well as accidental modifications. It uses AES-128 symmetric keys and is defined by RFC-4493. It is Federal Information Processing Standards (FIPS) approved.

Like any symmetric authentication algorithm, it takes a key and data as input and generates the authentication code. This is shown below:

$$\text{MAC} = \text{AES-CMAC}(k, m, len)$$

The inputs are:

k: 128-bit key

m: message to be authenticated

len: length of the message to be authenticated

The output is:

MAC: Message Authentication Code

This function may be implemented in either the Host or the Controller. If it is implemented in the Controller, the Host uses the HCI_LE_Encrypt command in order to generate the MAC. The HCI_LE_Encrypt command was explained in Chapter 9.

11.7.6 LE Secure Connections Confirm Value Generation Function f4

This function is used to generate the confirm values that are exchanged during the LE secure connections pairing process.

11.7.7 LE Secure Connections Key Generation Function f5

This is an intermediate function that is used to generate the derived keying values that would be used by function f6. It uses an AES-CMAC function with a predefined key to generate a key, T, and the same function once more to use that key T to generate the long term key (LTK) and MacKey.

The inputs of this function are random numbers generated by Master and Slave, BD_ADDR of Master and Slave and a shared Diffie-Hellman key.

The outputs of this function are LTK and MacKey.

11.7.8 LE Secure Connections Check Value Generation Function f6

This function is used to generate the check values during the authentication stage 2 in the pairing process.

The inputs of this function depend on the association model that is used (i.e., Just Works, Numeric Comparison, Out-Of-Band, or Passkey Entry).

The output of this function is a check value.

11.7.9 LE Secure Connections Numeric Comparison Value Generation Function g2

This function is used to generate the numeric comparison values during authentication stage 1 in the pairing process.

The output of this function is a 6-digit decimal value that will be used for numeric comparison.

11.7.10 LE Secure Connections Link Key Conversion Function h6

This function is used to convert a BR/EDR link key into an LE LTK or an LE LTK into a BR/EDR link key.

11.8 Pairing

The pairing process is used to establish the keys used to encrypt the link. Once the link is encrypted, the various keys to resolve the random address, verify signed data and encrypt future links are exchanged between the Master and the Slave.

Pairing is a three phase process. The first two phases are mandatory while the third phase is optional. The three phases are:

1. Pairing Feature Exchange.
2. Authentication and Encryption
   - Short-term key (STK) generation in case of LE legacy pairing.
   - Long-term key (LTK) generation in case of LE secure connections.
3. Transport Specific Key Distribution [Optional].

Phase 1 and Phase 2 may be performed on a link (encrypted or not encrypted) while Phase 3 is only performed on a link which is encrypted using the STK generated in Phase 2. These three phases are shown in Figure 11.4 and will be explained in detail in the next sections.

11.8.1 Phase 1: Pairing Feature Exchange

In this phase, the devices exchange their authentication requirements, capability information, key sizes, the keys that they can distribute, and the keys that they expect the remote side to distribute. This phase is also used in identifying which method of pairing they can use in the second phase. Pairing procedure is always initiated by the device in the Master role. The following is the set of features which are exchanged between the Master and the Slave during this phase.

11.8.1.1 IO Capability

The IO Capability provides information about the input and output capabilities of the device. It is used during the pairing feature exchange procedure (step 1 of the pairing process) to inform the IO capabilities of the device. Based on the IO capabilities of the two devices, the appropriate pairing procedure is selected in Phase 2 of the pairing process.

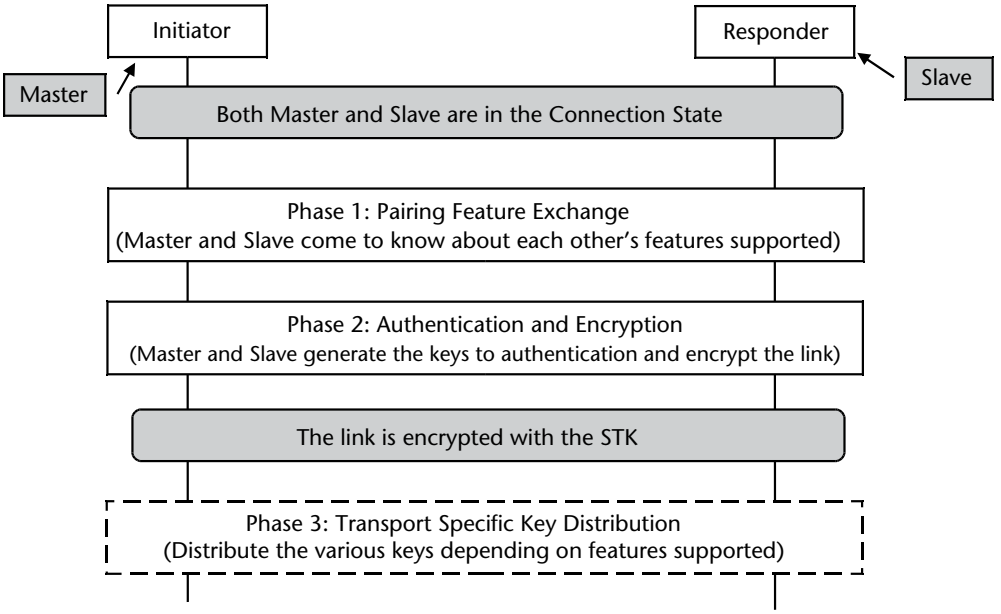

Figure 11.4 Three phase pairing process.

There are three possibilities with regards to the input capabilities of the device:

1. *No Input:* The device does not have any capability to take user input. An example of this could be a weighing machine which may not have buttons on it to take user inputs.
2. *Yes/No:* The device has some mechanism which can be used to indicate a 'yes' or 'no' input from the user. An example of this could be a key fob which has a couple of buttons which the user can use to indicate a 'yes' or a 'no'.
3. *Keyboard:* The device has a numeric keyboard which can be used to input the numbers '0' to '9' and two buttons to indicate 'yes' and 'no'. An example of this could be a remote control which has a numeric keypad which can be used to input the numbers '0' to '9'.

There are two possibilities with regards to output capabilities of the device:

1. *No Output:* The device does not have any display capability to display a 6-digit decimal number. An example of this could be a remote control which does not have display capability.
2. *Numeric Output:* The device has the ability to display or communicate a 6-digit decimal number to the user. An example of this could be a watch which can display the 6-digit decimal number to the user.

Based on the input and output capabilities of the device, the IO capability field is mapped as shown in Table 11.1. As an example, a device which has No Input capability and Numeric Output is termed as DisplayOnly since it can only display but it cannot take user input.

Table 11.1 IO Capability

	Output Capability	
Input Capability	*No Output*	*Numeric Output*
No Input	NoInputNoOutput	DisplayOnly
Yes / No	NoInputNoOutput	DisplayYesNo
Keyboard	KeyboardOnly	KeyboardDisplay

The IO capability of the two devices is used to decide the appropriate pairing method to use in Step 2 out of the following four possibilities:

1. Just Works.
2. Numeric Comparison.
3. Passkey Entry.
4. Out of Band (OOB).

11.8.1.2 OOB Authentication Data

In order to have additional security, an LE device may support the use of an external mechanism to generate data to authenticate the device. For example, an LE device may use the NFC protocol to generate authentication data. This provides additional security because the NFC protocol may involve touching the two devices and data exchange on a different protocol than LE. So the possibility of an intruder listening to that data is significantly reduced.

11.8.1.3 Encryption Key Size

LE supports an encryption key between 7 octets (56-bits) and 16 octets (128-bits). The higher the number of bits used, the higher the level of encryption. So the devices can select the encryption key size based on the resources available (memory, processing power) and the level of security needed. This value is exchanged between the two devices during Phase 1, and the lesser of the two values is used as the encryption key size for the next phases.

11.8.1.4 Repeated Attempts

This feature is similar to the support provided in BR/EDR for repeated attempts. This feature is invoked when a pairing procedure fails. If the pairing procedure fails, the device has to wait a certain interval before again trying to pair to a remote device or allowing a new pairing procedure from a remote device. This waiting interval increases exponentially with each failed attempt. The waiting interval keeps increasing with each failed attempt until the time it reaches an implementation defined maximum value. It helps to prevent an intruder from repeatedly trying the pairing procedure with several different keys.

For example, if an intruder device A tries to connect to device B. If the pairing procedure fails, then B sets the waiting interval to *i*. During this waiting interval *i* it

does not respond to any Pairing Request command or Security Request command from A. If A tries to pair after this waiting interval, and if the pairing procedure fails again, then B increases the waiting interval to 2 * *i*. If the pairing procedure fails a third time, the waiting interval increases to 4 * *i*. This makes it increasingly time consuming (and therefore difficult) for the intruder device A to try several different keys in order to pair to B.

11.8.2 Phase 2: Authentication and Encryption

After the Pairing Feature Exchange, both devices are aware of the capabilities of each other. Based on these capabilities, an appropriate paring method is selected. This can be one of the following four methods.

1. Just Works.
2. Numeric Comparison
3. Passkey Entry.
4. Out of Band (OOB).

The selection of the method is done as follows:

1. In LE legacy pairing, if both devices support the OOB pairing method, then the OOB pairing method is used.
2. In LE secure connections, if one or both devices support OOB pairing method, then the OOB pairing method is used.
3. If both devices have not set the MITM option, then the Just Works association model is used.
4. Otherwise, the IO capabilities of both the devices are used to determine the appropriate method.

In this phase, for LE legacy pairing, a Temporary Key (TK) is generated by each device based on the pairing method that is selected. This TK is used to generate the STK and encrypt the link. The formulae for generating TK and STK are explained in the Bluetooth specification.

The LE legacy pairing methods generate two keys, the temporary key (TK) and the short-term key (STK).

The LE secure connections pairing methods generate and use one key, the long-term key (LTK).

11.8.2.1 Just Works

This is the simplest pairing mechanism. It does not provide protection against eavesdropping and MITM attacks during the pairing process. Once the pairing process is over, this procedure provides security by using encryption.

11.8.2.2 Numeric Comparison

This method involves generating a 6-digit confirmation code on each of the devices. This confirmation code is presented to the users of the respective devices. If the

confirmation codes match, the users confirm this on their devices. Otherwise, the pairing process is aborted.

11.8.2.3 Passkey Entry

This method is used if one of the devices supports display capability and the other device supports keyboard capability. In this method, a 6-digit numeric passkey is displayed on one of the devices and the user is asked to enter that passkey on the second device. For example, if a TV is being paired with a remote, then the passkey will be displayed on the TV and the user will be asked to enter that passkey on the remote. This method provides protection against MITM attacks but limited protection against eavesdropping.

11.8.2.4 Out of Band (OOB)

This method is used if both the devices support the OOB Authentication Data feature. In this method the level of security depends on the security offered by the OOB mechanism used.

11.8.2.5 LE Secure Connections Pairing Phase 2

The detailed steps for LE Secure Connections Pairing Phase 2 are shown in Figure 11.5. The mains steps are:

1. Each device generates its own ECDH (Elliptic-Curve Diffie-Hellman) public-private key pair.
   - The key pair contains a private (secret) key and a public key. The public key (PK) can be shared with other devices while the private key (SK) is kept within the device and never goes out.
2. Pairing is initiated by each device sending its public key to the peer device.
3. Each device computes the Diffie-Hellman key (DHKey) as follows
   - DHKey = P256(SK of local device, PK of remote device)
4. Authentication Stage 1
   - Depending on the pairing method used (Just Works, Numeric Key Comparison, Out of band), the devices exchange the pairing information by using f4, f6 and g2 functions.
5. Authentication Stage 2
   - This stage confirms that both the devices have successfully completed the exchange.
   - Each device calculates the MacKey and LTK based on the keys exchanged in previous steps.
   - The initiating device then calculates a new confirmation value and transmits it to the responding device. The responding device checks the confirmation value.
   - The responding device then calculates a new confirmation value and transmits it to the initiating device. The initiating device checks the confirmation value.

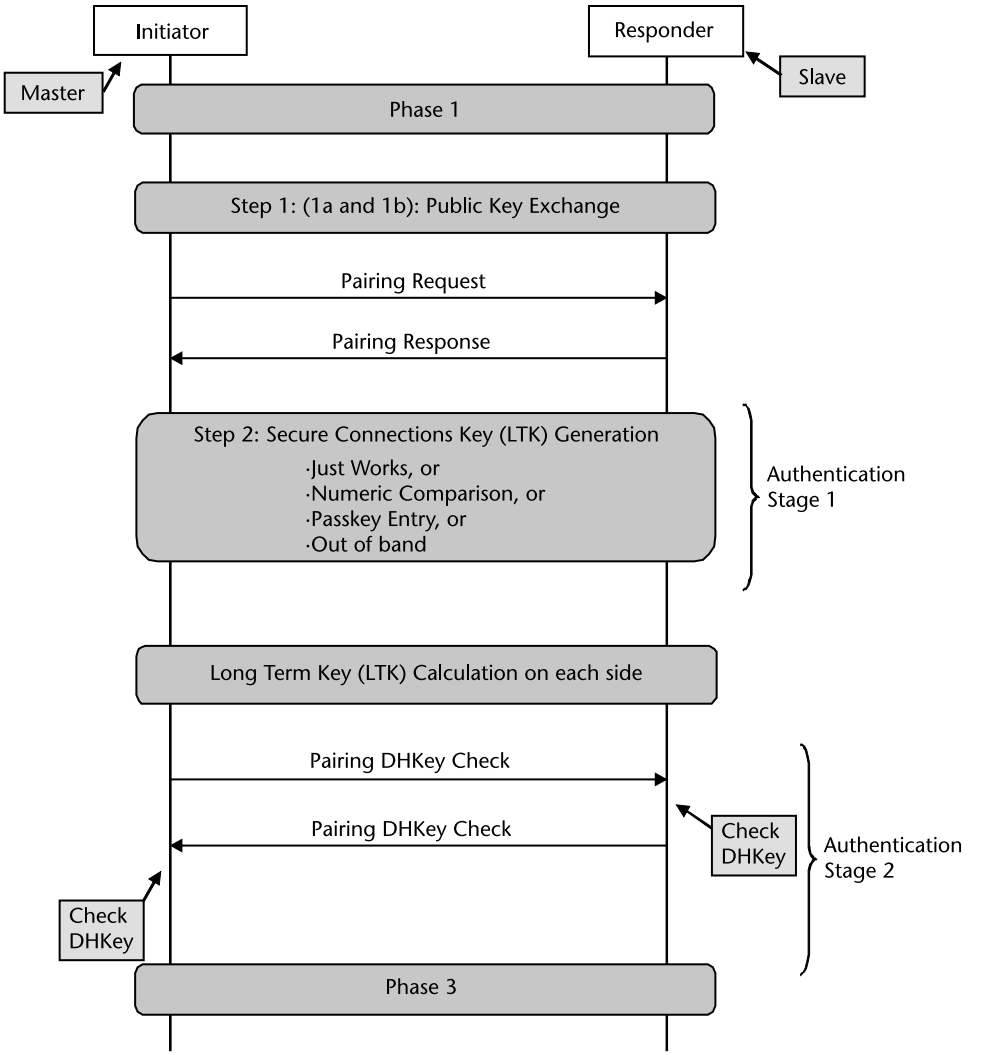

Figure 11.5 Pairing Phase 2 details for LE secure connections.

11.8.3 Phase 3: Transport Specific Key Distribution

This phase is optional and is performed only on a link which is encrypted using STK. In this phase, the Master and Slave distribute the keys to each other.

The various keys that are distributed in this phase for LE legacy pairing are:

- Long Term Key (LTK).
- Encrypted Diversifier (EDIV) and Random Number.
- Identity Resolution Key (IRK).
- Public Device Address or Static Random Address.
- Connection Signature Resolving Key (CSRK).

The various keys that are distributed in this phase for LE secure connections are the identity resolution key (IRK) and the connection signature resolving key (CSRK). These keys are briefly described below.

11.8.3.1 Long Term Key (LTK)

The Long Term Key (LTK) is a 128-bit key that is used to generate the key for an encrypted connection. The LTK is provided by the host to the controller on both the Master and the Slave side. The Master and Slave controllers use a combination of LTK, EDIV and Rand for LE legacy pairing and LTK in case of LE secure connection to encrypt the link.

11.8.3.2 Encrypted Diversifier (EDIV) and Random Number (Rand)

Encrypted Diversifier (EDIV) is a 16-bit stored value to identify the LTK. A new EDIV is generated every time a unique LTK is distributed. Random Number (Rand) is a 64-bit value used to identify the LTK. A new Rand is generated every time a unique LTK is distributed. A combination of LTK, EDIV and Rand is used by the Master and the Slave to encrypt the link.

11.8.3.3 Identity Resolution Key (IRK)

Identity Resolution Key (IRK) is a 128-bit key used to generate and resolve random addresses. Random address is a privacy feature that is introduced in LE. This allows a device to frequently use a different random address so that it's difficult to track that device. The random address can be resolvable or non-resolvable: This will be explained in detail in Chapter 14. The resolvable address is generated in such a manner that it can be resolved by the peer device if the peer device has the IRK along with the random address. In that case, the peer device will be able to identify the device transmitting data.

The IRK is used along with a random number to generate the random address. This random address is used for all further interactions with other devices. If a device wants a remote device to identify it, then it provides the IRK to that device. The remote device can then identify which device the packet is coming from by using a combination of the IRK and the random address. All other devices which don't have the IRK cannot identify which device is sending the packet and therefore cannot know if the random address belongs to the same device.

A Master that receives an IRK from a Slave can resolve the Slave's random device address. Similarly a Slave that receives an IRK from the Master can resolve the Master's random device address.

11.8.3.4 Connection Signature Resolving Key (CSRK)

LE provides the feature to sign the data. To sign the data, the sending device appends a 12-octet signature after the data PDU. The receiving device verifies the signature to check if the data is coming from a trusted source.

Connection Signature Resolving Key (CSRK) is a 128-bit key used to sign data and verify signatures on the receiving side. A different CSRK is used for each peer device to which signed data is to be sent.

11.9 Security Manager Protocol

The Security Manager Protocol (SMP) is used for pairing and key distribution. The L2CAP CID 0x0006 is used for all SMP commands. The format of the SMP commands is shown in Figure 11.6.

The Code field identifies the type of the command. The length and format of the Data field depends on the type of the command.

A timer of 30 seconds is used for all SMP procedures. If the timer expires, then the SMP procedure is considered to have failed and the higher layers are notified. If the higher layers need to restart the procedure, then a new physical channel needs to be established.

The various command codes supported by SMP are shown in Table 11.2. The command codes that have been introduced in specifications 4.2 to support LE secure connections are showin in italics in Table 11.2.

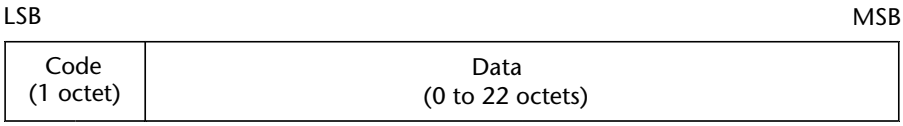

Figure 11.6 Format of SMP commands.

Table 11.2 SMP Command Codes

Code	*Description*	*Phase*
0x00	Reserved	—
0x01	Pairing Request	Phase 1
0x02	Pairing Response	Phase 1
0x03	Pairing Confirm	Phase 2
0x04	Pairing Random	Phase 2
0x05	Pairing Failed	Phase 2
0x06	Encryption Information	Phase 3
0x07	Master Identification	Phase 3
0x08	Identity Information	Phase 3
0x09	Identity Address Information	Phase 3
0x0A	Signing Information	Phase 3
0x0B	Security Request	Phase 1 (Used if Slave requests initiation of security procedures)
0x0C	*Pairing Public Key*	*Phase 2*
0x0D	*Pairing DHKey Check*	*Phase 2*
0x0E	*Pairing Keypress Notification*	*Phase 2*
0x0F – 0xFF	Reserved	—

11.9.1 Commands Used During Phase 1 (Pairing Feature Exchange)

In this phase, the Initiator (Master) and the Responder (Slave) exchange the information about paring features that they support so that the appropriate set of features can be used in Phase 2. The commands that are used during Phase 1 are shown in Figure 11.7. At the end of this phase, a pairing method is selected based on the Pairing Request and Pairing Response.

11.9.1.1 Security Request

This command is used by the Slave if it wants to request the Master to initiate security. In response to this command, the Master may send the Pairing Request command.

11.9.1.2 Pairing Request

The Pairing Request command is used to start the first phase of the pairing process which is Pairing Feature Exchange. The Initiator provides the set of features it supports in this command to the remote device. This includes the following:

- IO Capability: This indicates the IO capabilities of the device like DisplayOnly, KeyboardOnly, KeyboardDisplay etc. The different IO Capabilities were shown in Table 11.1.
- OOB Data Flag: This indicates whether Out Of Band data is available or not.
- AuthReq Flag: This indicates the requested authentication requirements. For example, whether bonding is requested or not or whether MITM protection is requested.

The various fields that are a part of this flag are as follows:

- Bonding Flag: indicates the type of bonding being requested.
- MITM: indicates whether MITM protection is requested.
- SC: indicates a request for LE secure connection pairing. This is a new flag introduced in specifications 4.2 to support LE secure connections.

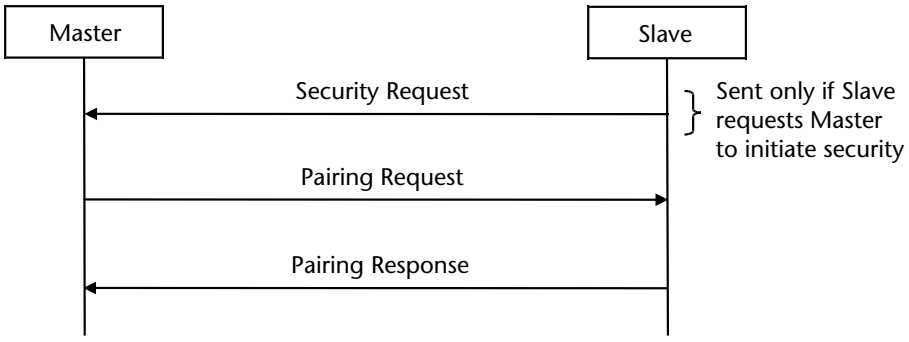

Figure 11.7 Sequence of commands used in Phase 1.

- Keypress: indicates that keypress notifications will be generated and sent to the remote side. This is a new flag introduced in specifications 4.2.
- Maximum Encryption Key Size: This indicates the maximum encryption key size that is supported by the device.
- Initiator Key Distribution: This field indicates which keys the Initiator is requesting to distribute in Phase 3 of the pairing process.
- Responder Key Distribution: This field indicates which keys the Initiator is requesting the Responder to distribute during Phase 3 of the pairing process.

The format for the Initiator Key Distribution and Responder Key Distribution field is shown in Figure 11.8. A Practical example of Pairing Request is shown in Figure 11.12.

11.9.1.3 Pairing Response

The Pairing Response command is used by the responding device to respond to the paring request. This command completes Phase 1 of the pairing process.

The different fields of this command are similar to the ones used in the Pairing Request command. In this command, the Responder confirms the exact Initiator and Responder keys to be used in Phase 3. If the Master had set any of the fields to zero in the Pairing Request command, then the Slave does not set them to one. Then, depending on its own capabilities, the Slave sets the remaining bits in the initiator key distribution and responder key distribution fields.

Once the Master receives the pairing response command, both devices know which keys will be exchanged in Phase 3. A practical example of pairing response is shown in Figure 11.11.

11.9.2 Commands Used During Phase 2 (Key Generation)

This phase is started after successful completion of Phase 1. After the Pairing Feature Exchange has completed, a paring method is selected for Phase 2. As mentioned earlier, there are four pairing methods that can be used:

1. Just Works.
2. Numeric Comparison.
3. Passkey Entry.

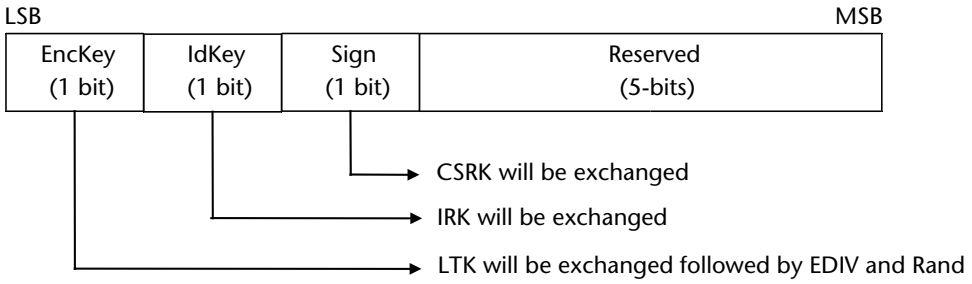

Figure 11.8 Format of initiator key distribution and responder key distribution.

4. Out of Band Pairing.

In this phase a short term key (STK) is generated and the link is encrypted using that STK in the case of LE legacy pairing. In the case of LE secure connections, a long-term key (LTK) is generated and the link is encrypted using that LTK. The commands that are used during Phase 2 for LE legacy pairing are shown in Figure 11.9.

11.9.2.1 Pairing Confirm

The pairing confirm command is used by both devices to send the confirm value to the peer device. This is a 128-bit value which is generated by the Master and the Slave based on the pairing method that is selected. The Master generates a 128-bit Mconfirm and sends it to the Slave and the Slave generates a 128-bit Sconfirm and sends it to the Master.

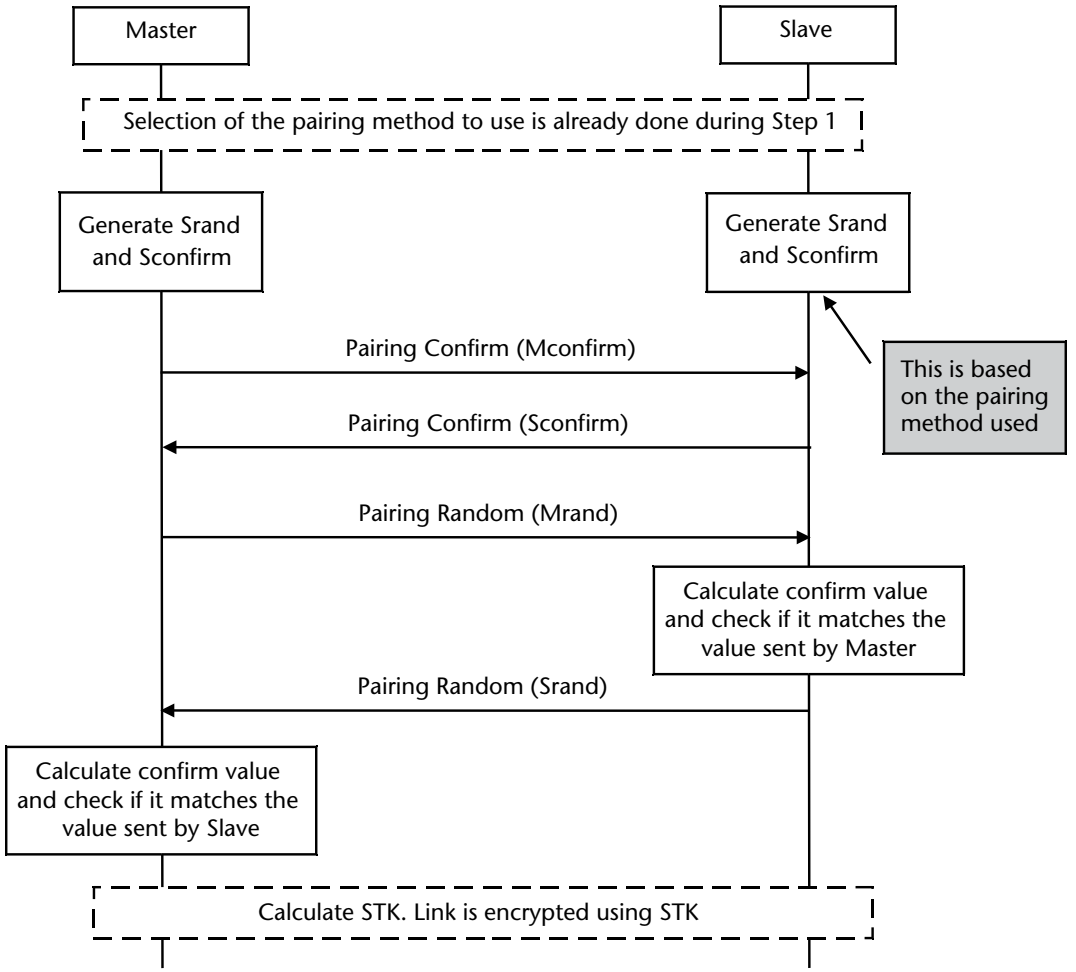

Figure 11.9 Sequence of commands used in Phase 2.

11.9.2.2 Pairing Random

The pairing random command is used by both devices to send the random number used in generating the confirm value that was sent in the previous pairing confirm command.

The Master sends the Mrand value to the Slave. The Slave calculates the confirm value based on this Mrand and checks if the value matches the value sent by the Master in the previous step (Mconfirm). If the value matches, then the Slave sends its own random number (Srand) to the Master. The Master does a similar calculation to check if the calculated value matches the confirm value that was sent by the Slave. After the Master verifies the confirm value, it calculates the STK and encrypts the link with the STK.

11.9.2.3 Pairing Failed

The pairing failed command is used if there is a failure during pairing. For example, if the confirm value received from the remote side does not match the confirm value calculated, then a Pairing Failed command is sent. A practical message sequence chart of these procedures is shown in Figure 11.12.

11.9.2.4 Pairing Public Key

This message was introduced in specifications 4.2 to support LE secure connections. It is used to transfer the device's own public key to the remote device. The initiator and the responder both use this method to provide their public key to the peer device.

The public key contains two parts referred to as X and Y coordinates. Each part is 32 octets in length.

11.9.2.5 Pairing DHKey Check

This message was introduced in specifications 4.2 to support LE secure connections. It is used to transmit the 128-bit DHKey check values generated using the LE secure connections check value generation function, f6. This message is used by both the initiator and the responder to provide the DHKey check value to the peer device.

11.9.2.6 Pairing Keypress Notification

This message was introduced in specifications 4.2 to support LE secure connections. It is used during passkey entry to inform the peer device that the keypad keys have been entered or erased. This method is used only by the device with KeyboardOnly IO capabilities.

The keypress notifications are used to indicate the following events to the remote side:

- Passkey entry started;
- Passkey digit entered;
- Passkey digit erased;

- Passkey cleared;
- Passkey entry completed.

11.9.3 Commands Used During Phase 3 (Transport Specific Key Distribution)

This phase is optional and started after successful completion of Phase 2. By the end of Phase 2, the link is encrypted and during this phase the transport specific keys can be distributed from the Master to the Slave and vice versa. The various keys that can be distributed are shown in Figure 11.10.

The keys are distributed first by the Slave and then by the Master in the sequence that is shown in Figure 11.10. Some or all of these keys may be distributed depending on what was agreed during Phase 1. The keys to be distributed during this phase were indicated in Phase 1 in the Key Distribution Field of the Pairing Request and Pairing Response commands. For example, the LTK is distributed from the Slave to the Master only if the Slave had earlier set the EncKey bit in the Key Distribution Field of the Paring Response command that it sent to the Master in Phase 1. (See Figure 11.8).

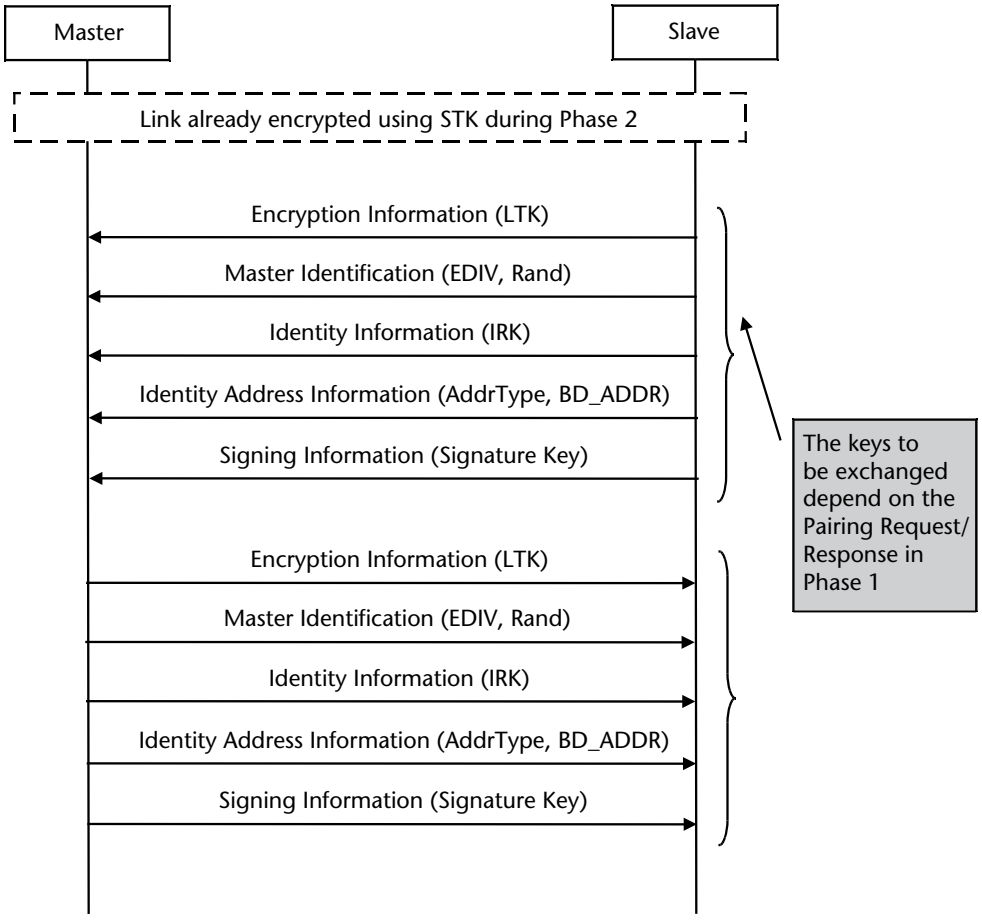

Figure 11.10 Distribution of transport specific keys in Phase 3.

11.9.3.1 Encryption Information

The Encryption Information command is used to distribute the LTK to be used for encrypting the connection. It contains the following parameter:

- LTK: 128-bit.

This command is only sent when the link has been encrypted with STK.

11.9.3.2 Master Identification

The Master Identification command is used to distribute the EDIV and Rand to be used for encrypting the connection. It contains the following parameters:

- EDIV: 16-bit.
- Rand: 64-bit.

This command is only sent when the link has been encrypted with STK.

11.9.3.3 Identity Information

The Identity Information command is used to distribute the IRK to be used for resolving the device. It contains the following parameter:

- IRK: 128-bit.

This command is only sent when the link has been encrypted with STK.

11.9.3.4 Identity Address Information

The Identity Information command is used to distribute the public device address or the static random address. It contains the following parameters:

- AddrType: 8-bit: To identify whether the address is a public device address or a static random address.
- BD_ADDR: 48-bit: The public device address or the static random address.

This command is only sent when the link has been encrypted with STK.

11.9.3.5 Signing Information

The Identity Information command is used to distribute the CSRK to be used for signing the data. It contains the following parameter:

- CSRK: 128-bit.

11.10 Practical Examples

11.10.1 Message Sequence for LE Legacy Pairing

Sample air sniffer captures of SMP transactions between an Initiator and Responder are shown in Figure 11.11 and Figure 11.12. The security procedures start at Frame #690 with a Pairing Request. There are two devices that are identified as Side 1 and Side 2. Side 1 is the Initiator and Side 2 is the Responder.

The parameters of the Pairing Request procedure are shown on the left side of Figure 11.11. The main points to note are:

1. IO Capabilities are set to KeyboardDisplay. This means that the device has a Keyboard and a Display.
2. MITM Protection has been set to Yes.
3. The Initiator Key Distribution specifies the keys that the Initiator will distribute. In this example the Initiator specifies the following:
   a. *Encryption Key:* Initiator shall distribute LTK followed by EDIV and Rand.
   b. *Id Key:* Initiator shall distribute IRK followed by its address.
   c. *Sign:* Initiator shall distribute CSRK.
4. The Responder Key Distributor specifies the keys that the Initiator requests the Responder to distribute. In this example, the Initiator has specified the following keys for the Responder to distribute.
   a. *Encryption Key:* Responder shall distribute LTK followed by EDIV and Rand.
   b. *Id Key:* Responder shall distribute IRK followed by its address.
   c. *Sign:* Responder shall distribute CSRK.

The Paring Response procedure is shown in Figure 11.12. The parameters of the Pairing Response are shown on the left side of the figure. The main points to note are:

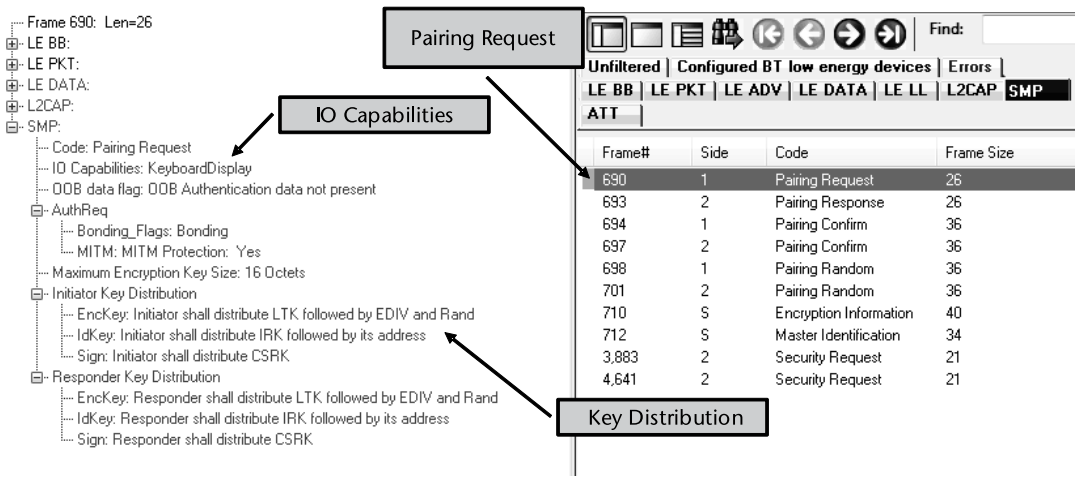

Figure 11.11 Example of Pairing Request.

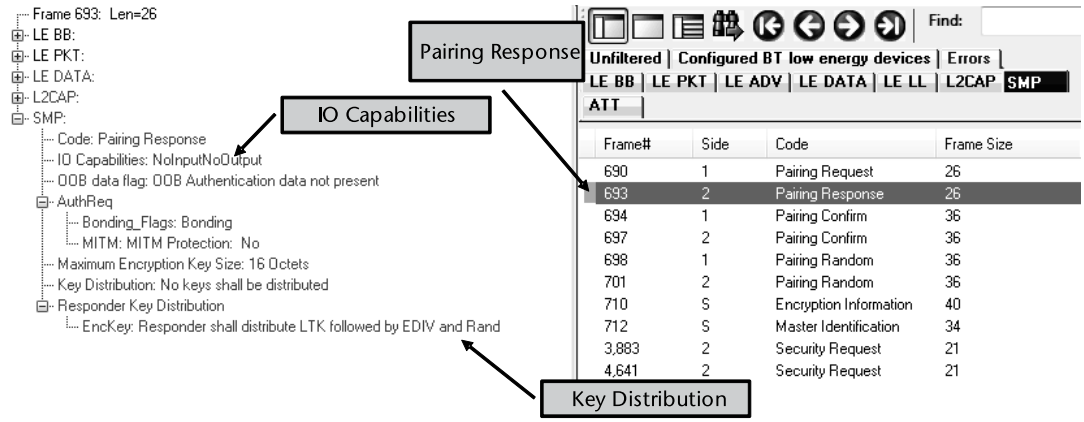

Figure 11.12 Example of Pairing Response.

1. IO Capabilities are set to NoInputOutput. This means that the device neither has a Keyboard nor a Display.
2. MITM Protection has been set to No.
3. The Responder Key Distribution specifies the keys that the Responder will distribute. In this example the Responder specifies only the following:
   a. *Encryption Key:* Initiator shall distribute LTK followed by EDIV and Rand.

Since one device supports KeyboardDisplay while the other supports NoInputNoOutput, the Just Works pairing method would be used.

The detailed message sequence chart corresponding to the air logs in Figure 11.11 and Figure 11.12 is shown in Figure 11.13. The SMP_M indicates the Master and SMP_S indicates the Slave.

The various phases of the pairing procedure are depicted as follows:

1. *Phase 1:* Phase 1 consists of the Master sending the Pairing Request and the Slave responding with a Pairing Response.
   a. Pairing Request from the Master specifies the IO Capabilities as KeyboardDisplay and the MaxKeySize to be 16 octets.
   b. Pairing Response from the Slave specifies the IO Capabilities of the Slave as NoInputNoOutput and the MaxKeySize as 16 octets.
2. *Phase 2:* Phase 2 consists of Pairing Random and Pairing Confirm exchanged between the Master and the Slave.
   a. The Master sends a Pairing Confirm which includes a 128-bit Mconfirm.
   b. The Slave responds with a Pairing Confirm which includes a 128-bit Sconfirm.
   c. After this the Master sends the Pairing Random with the Mrand value. The Slave calculates the confirm value based on this Mrand and checks if this matches the value sent by the Master in the Pairing Confirm Request.

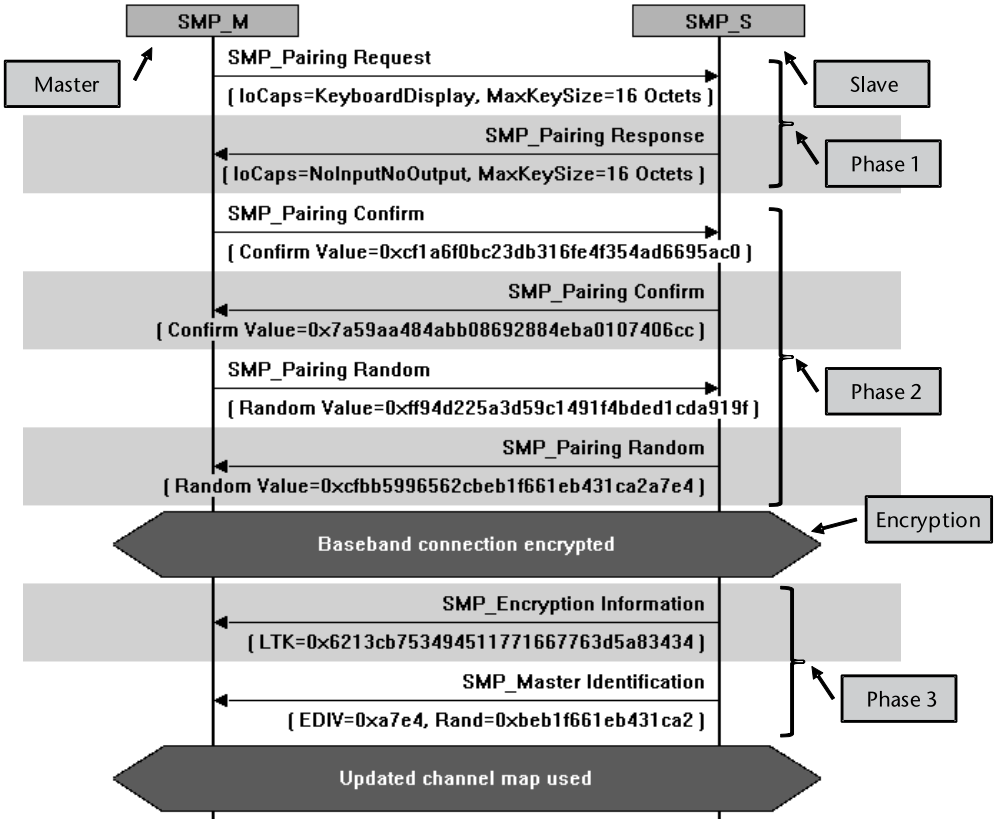

Figure 11.13 Example of SMP message sequence chart.

d. The Slave checks that the value matches and then sends the Pairing Confirm with its own Mrand value. The Master calculates the confirm value based on this Srand and checks if this matches the value sent by the Slave in the pairing confirm request.

3. *Phase 3:* After successful completion of Phase 2, the link is encrypted and the key distribution can start. The set of keys that will be distributed was specified in Phase 1.
   a. Encryption Information is used to distribute the 128-bit LTK.
   b. Master Identification is used to distribute 16-bit EDIV and 64-bit Rand.

11.10.2 Message Sequence for LE Secure Connections

Figures 11.14–11.16 illustrate the typical sequence of steps in setting up an LE secure connection.

11.10.2.1 Pairing Phase 1

The two sides involved in LE secure connections setup are termed as Side 1 and Side 2. Side 1 is the Master in this particular case. The sequence starts with Side 1