Security protocols in Bluetooth Standard

CT308A8800 – Secured Communication

December 2007

Sijan Shrestha – 0316010
Cheikhoul Seck – 0331024
Were Oyomno - 0315972

1. Introduction

Bluetooth is an open standard for wireless connectivity with industry backers mostly from the personal computers (PC) and mobile phone industries. Not surprisingly, its primary market is for data and voice transfer between communication devices and PCs. In this way, it's similar in purpose to the IrDA protocol. Bluetooth, however, is a radio-frequency (RF) technology utilizing the unlicensed 2.4GHz Industrial-Scientific-Medical (ISM) band. Target applications include PC and peripheral networking, hidden computing, and data synchronization such as for address books and calendars. Other applications could include home networking and home appliances of the future such as smart appliances, heating systems, and entertainment devices.

It was initially developed by Ericsson but is formalized as an industrial standard by the Bluetooth Special Interest Group (SIG). The SIG was formed by Ericsson, Intel, Toshiba, Nokia, and IBM but is now expanded to include about 1800 members.

2. Brief Overview

The communication in Bluetooth is done over a radio communication band at 2.4GHz. This band is divided into several frequencies, (23 or 79 which depends on the region). Each second is divided into 1600 timeslots of length 0.625ms. Each timeslot can contain up to ca 620 bits. The frequency is changed for every timeslot, i.e. 1600 times per second. This has some positive effect on security while it makes it more difficult to follow the communication between Bluetooth units. For a skilled attacker this frequency hopping is not an effective obstacle [3].

Packets are entities of information that are sent over the channel. Packets can span one, three, or five timeslots. The units use these packets to send information packets to each other.

Access Code(68)	Packet Header(58)	Payload (0-2745)

Figure 1: Bluetooth generic packet format

The access code is used for synchronisation. The access code and the packet header are never encrypted during the data transmission.

The protocol stack makes up the core portion of the Bluetooth implementation. This stack enables devices to locate each other and establish a connection. Through this connection, devices can exchange data and interact with one another through various applications

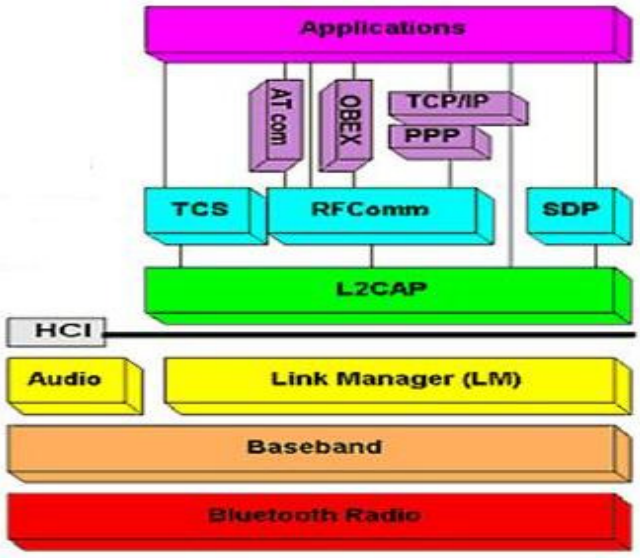

Figure 2: Illustration of Bluetooth's protocol stack

The baseband layer and the link control layer enable the physical Radio Frequency (RF) link between Bluetooth devices. Two kinds of physical links can be formed with their corresponding baseband packets: Synchronous Connection-Oriented (SCO) and Asynchronous Connectionless (ACL).

The Link Manager Protocol is responsible for setting up the link between Bluetooth devices. This includes security aspects like authentication and encryption by generating, exchanging, and checking of link and encryption keys.

The Logical Link Control and Adaptation Protocol (L2CAP) provides both connection-oriented and connectionless data services to the upper layer protocols with protocol multiplexing capability, segmentation and reassembly operation, and group abstractions. L2CAP is defined only for ACL links and not supported for SCO links, as specified by the Bluetooth Specification.

The Service Discovery Protocol (SDP) is used to get specific information about a remote device, such as available services. Using SDP characteristics of these services can be queried.

The RFCOMM protocol provides emulation of serial ports over the L2CAP protocol. The protocol is based on the ETSI standard TS 07.10. Only a subset of the TS 07.10 standard is used, and some adaptations of the protocol are specified in the Bluetooth RFCOMM specification.

The Bluetooth Specification defines a set of usage models or scenarios for using Bluetooth enabled radio devices. These usage modules, referred to as profiles, define standard ways of communicating with Bluetooth-enabled devices. It is intended to reduced the risk of interoperability issues between different manufacturers and provide a basis for the development of a common set of functions that Bluetooth enable devices will support. A profile does not need to include all the protocols defined in the Bluetooth protocol stack. It can be a subset of various protocols in the stack.

3. The Security Architecture

Right from its inception Bluetooth was designed with security in mind, this evident in the security requirement for even the simplest uses. This security is implemented through symmetric-key mechanisms for; authentication, link encryption and key-generation.

Authentication in Bluetooth verifies the devices are communicating with devices they should be, this necessitates the link-key, $\boldsymbol{K}$. The link key, $\boldsymbol{K}$ is not used directly in encryption, rather a ciphering key, $\boldsymbol{K_C}$ derived from further computations with $\boldsymbol{K}$ is used. The $\boldsymbol{K_C}$ is also changed after each transmitted package.

Security functionality in Bluetooth provides a system with basic enough building blocks to setup a private radio link between two devices. The Bluetooth SIG white paper outlines how to handle requirements introduced by different security modes [11].

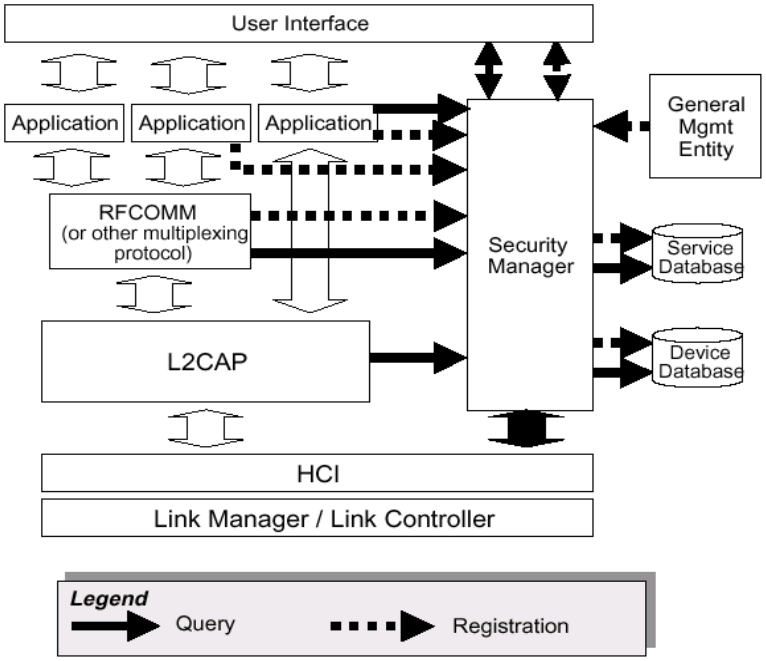

Figure 3: Bluetooth's security architecture [3, 11, 12].

Naturally, different services have different security requirements, i.e. authorization, authentication and confidentiality. It is the aim of an access control mechanism to avails a way for applications to request the type of connection they want i.e. service level enforced security.

Devices with established security relation lasting beyond the current session are known as trusted devices e.g. phones, headsets, PDA's one would like to connect to on numerous occasions. Trusted devices have unconditional access to all services running on the host after their identity has been confirmed via the authentication protocol. Alternatively it is possible to have relations in which trust is defined per service/group of services rather than per device.

Services also have their own security requirements based on the basic rules of link level security e.g. a service cannot request encryption without authentication. Thus the three basic link level securities that need to be obeyed are:

1. Authorization and authentication
2. Authentication only
3. No Authorization or Authentication – implying services open to all devices

Flexibility is desired to provide individual setting of access policies of different services e.g. opening up services for one application does not automatically also open up the service for other applications. This should be achieved while keeping the user interaction to a minimum.

In Bluetooth protocol multiplexing may occur above L2CAP layer as illustrated in Figure 2. E.g. in RFCOMM, OBEX etc. Thus the architecture accounts for this by considering that different protocols enforce different security policies for different services. The lower layers remain oblivious of the upper layers security policies. The Bluetooth security implementation as illustrated in Figure 2. is a challenge-response system using the pass-key as the secret key. The Security Manager (key unit) main goals are:

1. Stores services security information - Service Database
2. Stores devices security information - Device Database
3. Accept/Reject connection requests by Protocols/Applications
4. Enforces authentication and/or encryption prior connection setup
5. Initiate setup of trusted relationships with device
6. Prompt user/application for pass-key when required

The security manager does these tasks on applications behalf; hence from the security managers perspective devices are defined in three trust levels:

1. Trusted devices – previously authenticated having a known and stored link key labeled trusted
2. Untrusted devices – previously authenticated having a known and stored link key labeled untrusted
3. Unknown device – no security information exists for it

Trusted relationships are established via pairing procedures, with possibility of moving devices between trust levels. Thus whenever a device has an associated link key, authentication is done as prescribed by LMP and baseband specifications. To be verified as trusted the device has to pass the authentication stage and its flag changed to trusted. The security manager also maintains a service database Figure 2. This database controls settings for various services, defined by three attributes:

1. Authorization – trusted device granted access while untrusted devices need to authenticate themselves first.

2. Authentication – remote device need to be authenticate prior to accessing the service
3. Encryption – link is first switched to encryption mode before granting access to services/

Services attributes are distinguished for incoming and outgoing connections. It is general practice for each application to register its service with the service manager and define desired security level. However if no service database exists for a particular incoming/outgoing connection, default setting used; incoming connections need to be authorized implying also authentication and for outgoing connections only authentication is needed.

Channel establishment is the creation of a logical connection between two peers. At the L2CAP layer, it is characterized by their channel identifiers (CID). A channel serves a single application/higher level protocol. The desired security enforcements are known after determination of what security measures are requested by the service. The typical L2CAP connection is illustrated in Figure 4. In general this proceeds as follows:

1. Connection makes request to L2CAP
2. L2CAP requests access from security manager.
3. Security manager looks up services security policy in service database
4. Security manager looks up security policy for device in device database
5. If necessary, security manager enforces authentication and encryption
6. Security manager grants access to the service
7. L2CAP continues to establish the connection

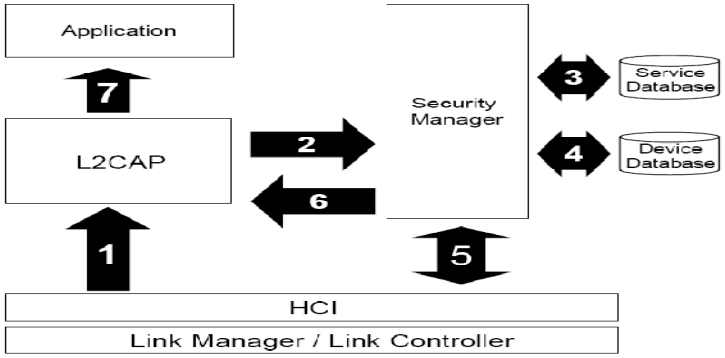

Figure 4: Access control procedure of an accepted L2CAP channel establishment [12].

For incoming connection request first the L2CAP layer queries the security manager with BD_ADDR and the protocol, based in this the security manager via the databases then decides whether to grant/deny and if authentication/authorization/ encryption is needed. To avoid duplicated request the security manager, stores temporary information concerning the status of the request. This scenario of duplicate/triplicate request is also common in outgoing connections thus same mechanism is used.

4. Algorithms, Ciphers and Encryption Engines

Bluetooth uses two main ciphering algorithms; block and stream ciphers. These are a 128-bit block SAFER+ based cipher and a 132-bit stream Ciphering Algorithm (E_0). Bluetooth's custom algorithms based on SAFER+ are the Unit Key Algorithm (E_{21}), the Initial Key Algorithm (E_{22}), the authentication algorithm (E_1). The algorithm based on stream cipher is the Encryption Algorithm (E_3). Block cipher was chosen because it lends to a natural transformation of the used 128-bit data block with a 128 bit key which can be directly used as data input as well.

4.1 SAFER+

SAFER+ is descendant of Secure and Fast encryption Routine (SAFER), block cipher with corrected key scheduling increased rounds, increased block size (64 to 128 bits) and introduction of the Armenian Shuffle permutation with a diffusion of a single-bit modification in the input data. SAFER+ has 16 rounds and like its predecessor it uses the round construct consisting of pseudo-Hadamard transformations, substitution tables and sub-key insertion. SAFER+ has 2 subsystems:

1. Encryption Subsystem:
2. Key Scheduling subsystems: This subsystem provides the round key for each encryption round in the encryption subsystem. Each round key consists of 2 vectors of 16 octets. However the last round key K_{17} is a single 16 octet vector that is added to the output of the last round. The rounds are input into the SAFER+ round mechanism where they are added into the round data by intertwined modulo 256 and XOR additions.

In reality Bluetooth uses a modified version of SAFER+ algorithm called E_1 Algorithm; the modification is that the original input to algorithm is added to the input of the third round thus making E_1 into a non invertible mapping. The reasoning behind this modification was to prevent the algorithm from being used for encryption thus avoiding export regulation problems with US government.

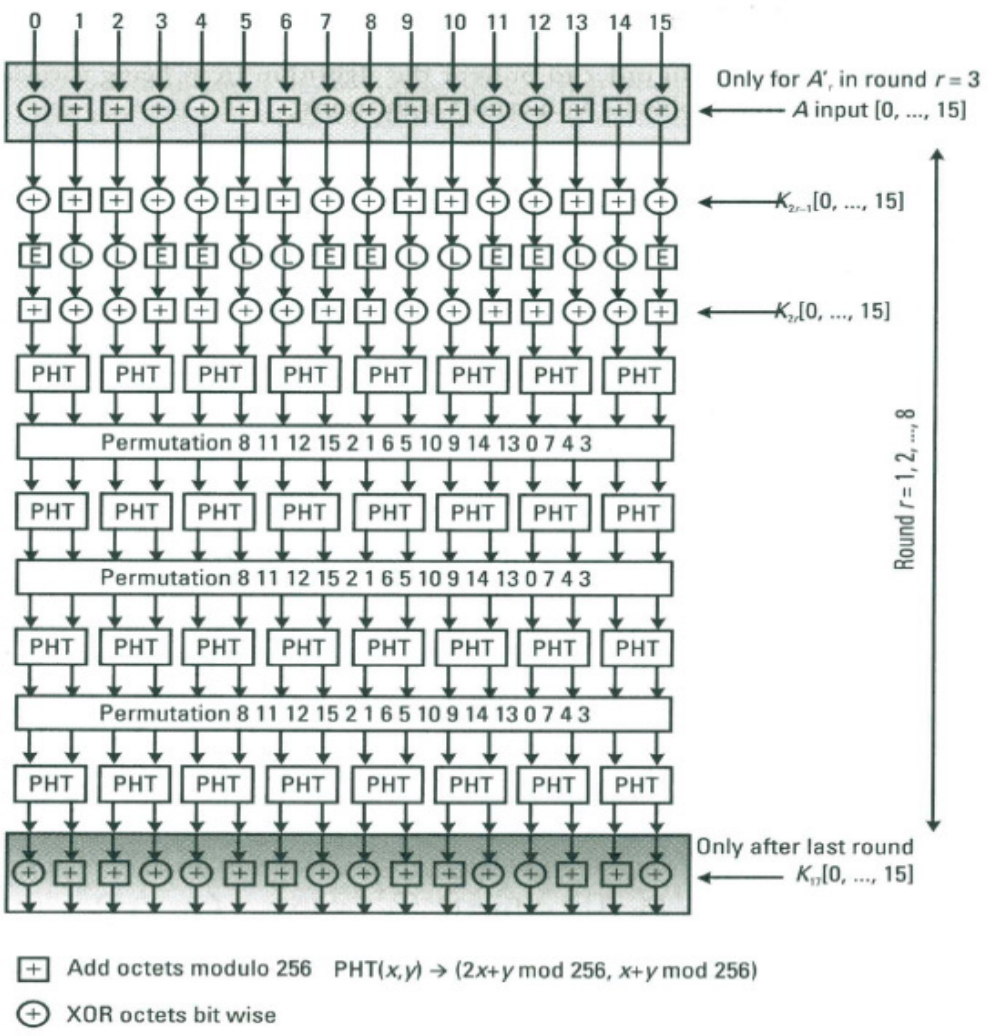

Figure 5: Single round of E_1, the modified SAFER+ algorithm used in Bluetooth [3].

4.2 Authentication Algorithm (E_1)

The Bluetooth's Authentication algorithm (E_1) is also known as the Message Authentication Code (MAC) Algorithm. It is based on SAFER+ as discussed in previous subsection.

$$SAFER+: \{0,1\}^{128} \times \{0,1\}^{128} \rightarrow \{0,1\}^{128}$$
$$(k, x) \mapsto y = SAFER+ (key = k, input = x)$$

E_1 is thus defined as follows

$$E_1 : \{0,1\}^{128} \times \{0,1\}^{128} \times \{0,1\}^{48} \rightarrow \{0,1\}^{32} \times \{0,1\}^{96}$$
$$(K, RAND, address) \mapsto (SRES, ACO) \quad ; SRES \text{ and } ACO \text{ hash } (\mapsto) \text{ derivatives}$$

The SAFER+ algorithm is used twice with different keys, firstly as SAFER+ and secondly as E_1 in the third round.

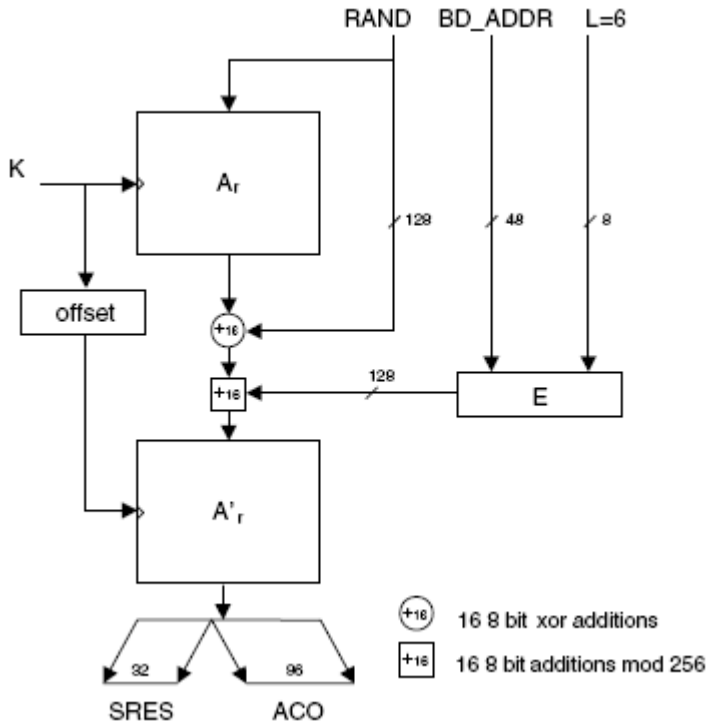

Figure 6: E1 construction using the modified SAFER+ block cipher

4.3 Unit Key Algorithm (E_{21})

E_{21} is based on the modified SAFER+, E_1 and is formally specified as:

$$E_{21}: \{0,1\}^{128} \times \{0,1\}^{48} \rightarrow \{0,1\}^{128}$$

Since E_1 not SAFER+ algorithm is used, E_{21} cannot be directly used as an invertible encryption algorithm.

4.4 Initial Key Algorithm (E_{22})

This algorithm is also based on E_1 and only slightly differs from E_{21}. It is formally specified as:

$$;(PKEY', RAND, N') \mapsto E_1(X, Y)$$
$$:X = \cup_{i=0}^{15} PKEY'[i \bmod N']$$

$$; Y = RAND[0 \dots 14] \cup (RAND[15] \oplus N')$$

4.5 Encryption Key Algorithm (E_3)

The encryption key K_C, is not directly used in the encryption rather is used to generate the constraint key that is input into the E_0 algorithm.

$$E_3 : \{0,1\}^{128} \; x \; \{0,1\}^{128} \; x \; \{0,1\}^{96} \rightarrow \{0,1\}^{128}$$
$$; (K, RAND, COF) \mapsto hash(K, RAND, COF, 12)$$

4.6 Ciphering Algorithm (E_0)

It is mainly composed of 4 Linear Feedback Shift Registers (LFSR) and a FSM as combined circuit. The FSM introduces nonlinearity thus making it difficult to re-compute the initial state via observation of key stream data. The LFSR's are fully characterized windmill polynomials with a degree sum of 128 as follows:

$$LFSR_1: \; f_1(t) = t^{25} + t^{20} + t^{12} + t^8 + 1$$
$$LFSR_2: \; f_2(t) = t^{31} + t^{24} + t^{16} + t^{12} + 1$$
$$LFSR_3: \; f_3(t) = t^{33} + t^{28} + t^{24} + t^4 + 1$$
$$LFSR_4: \; f_4(t) = t^{39} + t^{36} + t^{28} + t^4 + 1$$

The output sequence *X1=(X10, X11 …)* of $LFSR_1$ with an infinitely long clock is thus be formally expressed as $X_1(t) = \sum_{i=0}^{\infty} x_{1i} t^i$ thus represented as $X_1(t) = \frac{g_1(t)}{f_1(t)}$. The polynomial divisions are by ordinary polynomial arithmetic but using with, modulo 2 arithmetic', in the coefficients.

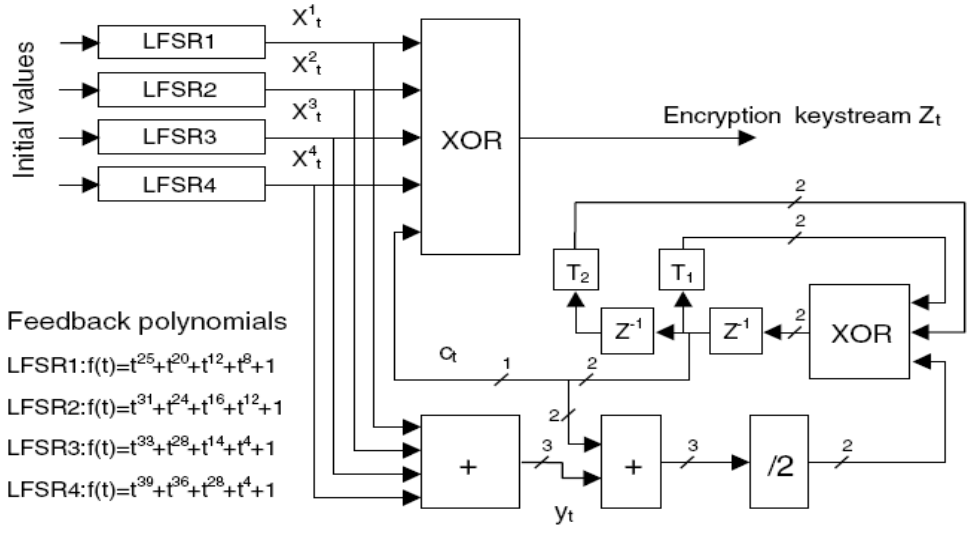

Figure 7: E_0 Stream cipher illustration.

In practical use one has to decide how the algorithms' should be implemented i.e. in software or hardware. The latter offer considerable advantages when speed and power usage are vital, while the former capitalize on flexibility and fast development time. In the Bluetooth case SAFER+ and its derivatives were implemented in software and $\boldsymbol{E_0}$ in hardware. This decision was influence by low power requirements.

5. Key Management

The Bluetooth system has various phases and modes of operation. The two distinct phases are the phase where two units want to communicate without having contact before and the phase where this communication takes place after prior communication. It's obvious that, in the former case the key has to be generated for the first time while in the latter case the communication units can use the secret keys from the previous communication. Different management schemes should be used to manage the keys in different phases.

The security of Bluetooth is basically based on symmetric key cryptography in which a private key called link key is shared between two or more parties. The link key can be semi permanent with a life time having more than one session, or it can be temporary where its lifetime is limited to only one session. The link keys are used for authentication and generation of encryption keys which is denoted by K_C . The management requirements of Bluetooth keys suggest four different types of keys;

1. The combination key K_{AB}
2. The unit key K_A
3. The temporary key K_{master} and
4. The initialization key K_{init}

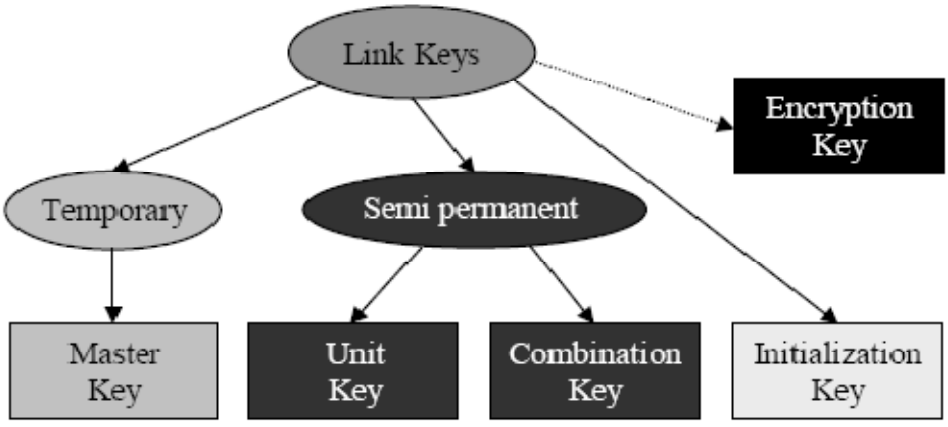

Figure 8: Bluetooth key structure [11]

Functionally the unit key and combination key plays the same role but are different in a way they are generated. The combination key K_{AB} is generated by the involvement of two units A and B whereas the unit key K_A is generated with a single unit. Basically the unit key is generated

during an initialization phase. The unit key is basically used when there is a little memory in the unit using it. Combination keys are used for more security requirement of the units.

The master key K_{master} is typically used to broadcast messages. When one unit (so called master) wants to communicate with more than one slave simultaneously using the same encryption key, the master key is used. It is basically a temporary key that replaces the current link key.

The initialization key K_{init} is used during the initialization phase when there are no unit and combination keys. The generation of this key requires a Personal Identification Number (PIN) code. This PIN can be a number provided with the Bluetooth unit or can be selected arbitrarily by the user and entered in both units that want to meet.

Now we will look how the keys are generated. Different algorithm like E_{21} and E_{22} are used to generate the keys. We have discussed about these algorithms in previous section.

5.1 Creation of Unit key K_A

Unit key is first created when Bluetooth device is first turned on which is unique for every device. Unit key is only used when the device has less memory to store the session keys. The Unit key is generated using E_{21} algorithm and is based on [12]:

1. Random number RAND, generated by the device, and
2. The Bluetooth device address ADDR

The following figure shows the generation of Unit key in devices A and B:

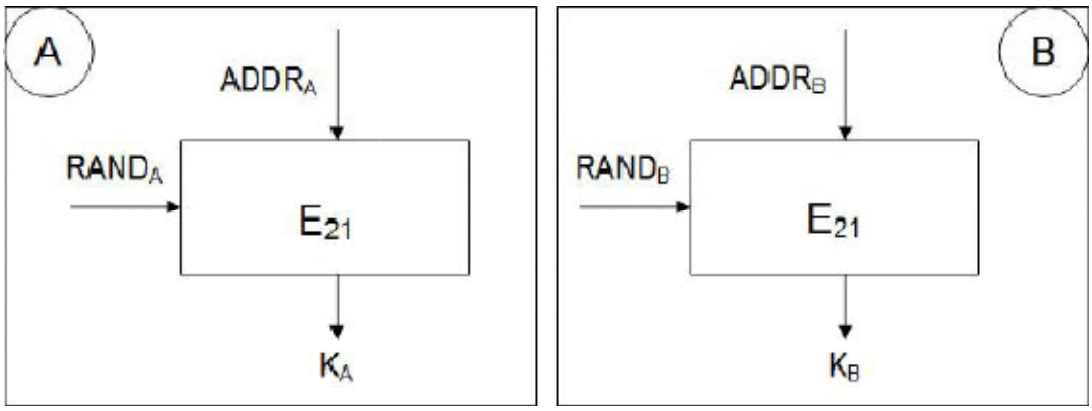

Figure 9: showing generation of Unit key

5.2 Creation of K_{init}

The initialization key is generated using the E_{22} algorithm. The inputs to this algorithm are:

1. A BD_ADDR
2. The PIN code and its length
3. A 128 bit random number IN_RAND

The following figure shows how K_{init} is generated using E_{22} algorithm. The PIN code is available to both the communication devices.128 bit IN_RAND is transmitted in a plain text. Critical issue here is about BD_ADDR .If one of the devices has a fixed PIN; they use the BD_ADDR of the peer device. If both have a variable PIN, they use the PIN of the slave that receives the IN_RAND. In the following figure the BD_ADDR_B is used where both devices have variable PIN.

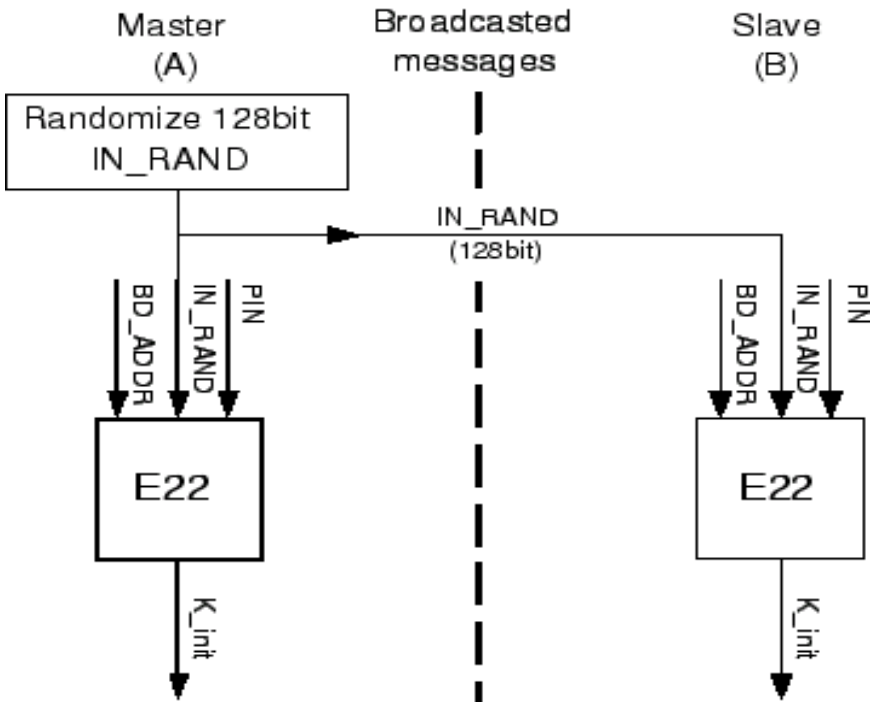

Figure 10: Generation of Initialization key [13].

The initialization key generated is used only during the pairing process and after creation of link key (K_{AB}), the initialization key is discarded.

5.3 Creating of K_{AB}

Once the initialization key is created, the devices create the combination key $K_{AB.}$ The combination key is created by the two inputs to the E_{21}algorithm:

1. A BD_ADDR
2. The 128 bit random number LK_RAND

Each units generate its own LK_RAND and transfer it to the other unit xoring it with the initialization key K_{init}. Since both devices knows K_{init} ,again xoring it with LK_RAND send by other unit, gets the LK_RAND of the other unit. In this way, each device has both random numbers LK_RAND_A and LK_RAND_B .

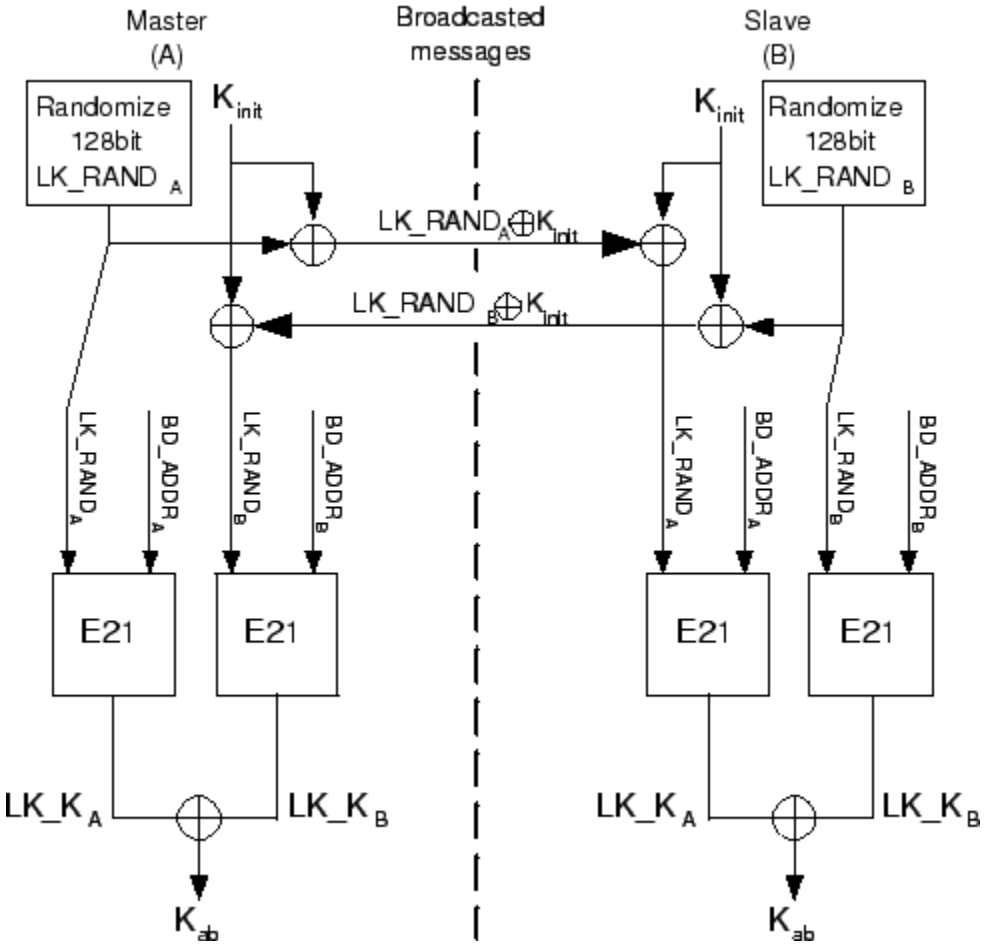

Figure 11: Showing the generation of combination key [13].

As shown in the figure, the E_{21} algorithm is done twice in each side to get two resulting values LK_K_A and LK_K_B. Xoring these two resulting values generates the combination key on the both sides.

5.4 Mutual Authentication

After having the link key K_{AB}, the mutual authentication is performed which is based on challenge-response scheme. During the authentication process, one unit is designated as the

verifier and the other unit as the claimant that has to prove to the verifier that it knows the same secret, the current link key that the verifier has. The verifier generates a 128 bit random number called AU_RAND_A and sends as a plain text to the claimant. Claimant calculates a 32 bit SRES using an algorithm E1 and send it to the verifier. The verifier also calculates SRES its side using E1 algorithm. Now,if both calculated SRES are same, the response work is successful and the verifier and claimant change roles and repeat the entire process. The following figure shows the process of mutual authentication [13]. The inputs to E1 are:

1. The random word AU_RAND_A
2. The link key K_{AB}.
3. Its own Bluetooth devices address (BD_ADDR_B).

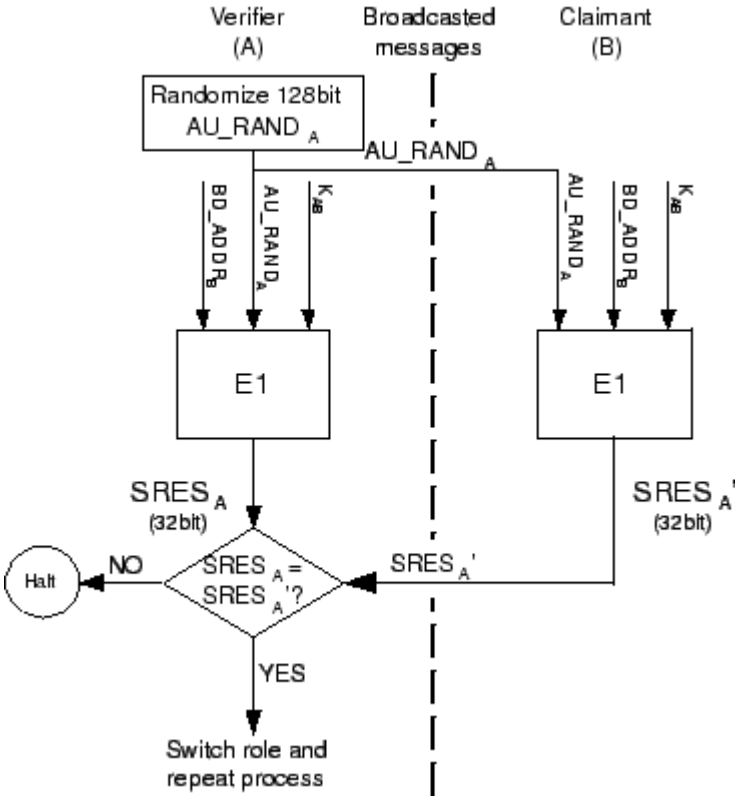

Figure 12: Mutual authentication process [13].

As the side-effect of a successful authentication procedure an auxiliary parameter, the Authenticated Ciphering Offset (ACO) is computed. This is used to create the encryption key.

5.5 Creation of Encryption Key (K_C)

Encryption key can be generated after a successful generation of the link keys. Encryption key can be generated by using E_3 algorithm which takes the inputs [12]:

1. EN_RAND_A ,generated by the device A
2. K_{link} ,link key generated, and
3. COF, Ciphering Offset Number which is the ACO value which is generated during the mutual authentication protocol.

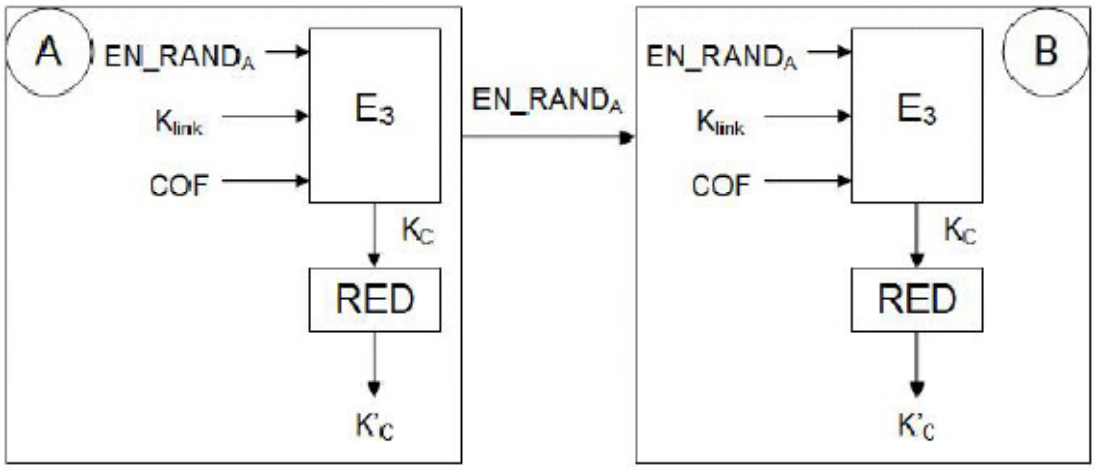

Figure 13: Generation of encryption key [12].

The encryption key K_C by the algorithm can be reduced in length to form the encryption key K'_C.

5.6 Generation of the key stream

The key stream K_{cipher} is generated using the encryption key K_C, Bluetooth address and the clock of the master and using the encryption algorithm E_0.The following figure shows how it is generated [12]:

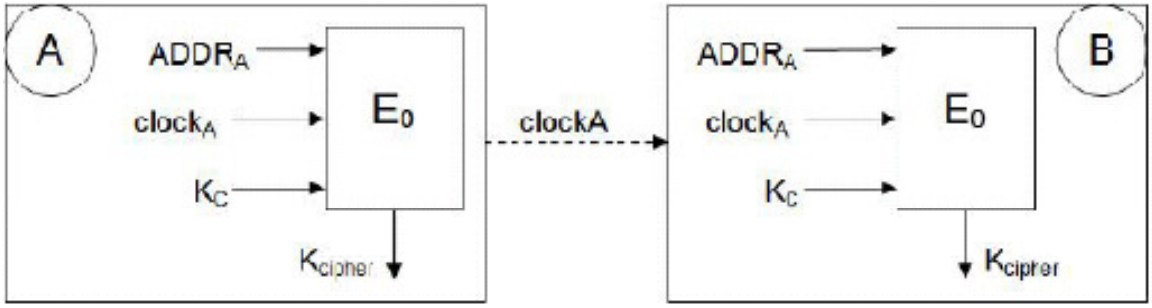

Figure 14: Generation of key stream[12].

The key stream K_{cipher} is xored with the data that has to encrypt as shown in the following figure:

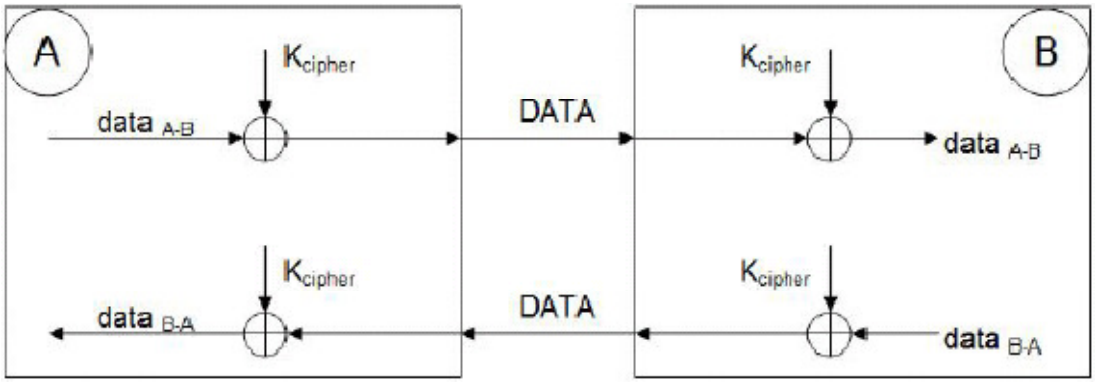

Figure 15: showing data encryption [12].

6. Strengths and Weaknesses

Essentially as a result of a thorough thought, Bluetooth security and the scope of potential attacks have been minimized more so when it involves obtaining the link key, this especially in the case if the SIG's recommendations are strictly followed. Thus security compromises will result either when the security is not set or via some erroneous implementation. However, there still exist few known mechanisms to bypassing Bluetooth's security measures.

6.1 Eavesdropping - data confidentiality

Obviously when Bluetooth connection has been setup without link encryption it is easy for other to listen on this communication and even replace the payload data. However when the link encryption is activated the listening in is eliminated, but the payload substitution may still occur by careful manipulation of data and CRC. This means that the confidentiality is thus pinned on the cipher strength $\boldsymbol{E_0}$ e.g. assuming the attacker has acquired the payload key $\boldsymbol{K_P}$, after determining the constraint key$\boldsymbol{K_C'}$, which for simplicity is assumed to be constant during the session duration. Also assuming they have access to a plain text as long as the cryptanalysis done to recover $\boldsymbol{K_P}$. However the considering time constraint its evident that the time amount it will take for the eavesdropper to determine $\boldsymbol{K_P}$ places a lower limit on the time to determine $\boldsymbol{K_C}$ meaning the time to determine $\boldsymbol{KC}$ is prohibitive to launch meaningful attacks as one has to reverse engineer it from one to several $\boldsymbol{K_P}$ and on releasing the assumption of fixed $\boldsymbol{K_C}$ this becomes a very difficult if not impossible feat.

6.2 Impersonation

In as far as this type of attack is eminent there are 2 alternatives an attacker could utilize:

1. Impersonate the legitimate sender/receiver: - To be successful the attacker must correctly answer the authentication challenge given to the receiving unit and since we already saw that there is currently no attack that can achieve this on E_1within realistic time that means this attack is futile.
2. Replace/insert payload/data sent: - This is more feasible especially when there is no link encryption as it only requires the correct setting of the CRC after altering the payload. When encryption is turned on this attack is still feasible as the ciphering basically consists of adding modulo 2 of the key bits stream to the data i.e. a linear operation like the CRC thus possible for the attacker to compute how to the CRC ought be modified to agree with the data. This means that the only way that the modifications on the data can be detected is if the CRC is not altered to agree with the modification

From the above scenarios, it is evident that it is only possible to alter the payload data when link encryption is not activated. However when link encryption is activated this attack is basically only reduced to communication disruption as the attacker will not be able to add meaningful

data. In that sense this could be concluded as a successful Bluetooth if the attackers aim is to disrupt the communication.

6.3 Pairing attacks

This has been demonstrated by many scholars to be vulnerable especially if the attacker is present at the pairing time. Typically two types of pairing attacks exist, active and passive pairing attacks on keys (combination/unit keys).

The combination key attack proceeds as follows. Given that in the pairing instance the current key used is K_{INIT} where $K_{INIT} = E_{22}$ ***(BD_ADDR$_A$, IN_RAND, PKEY). IN_RAND*** is sent to unit ***B*** in plaintext, thereafter K_{INIT} is then used to encrypt random values LK_RAND_A and LK_RAND_B which are used to compute K_{AB}. Thus is the attacker observes all this exchange then, it then summarizes that for him to attack device ***A*** he will need ***IN_RAND,*** $K_{INIT} \oplus LK\_RAND_A$, $K_{INIT} \oplus LK\_RAND_B$, ***AU_RAND*** and ***SRES.*** It is only the pass-key that the attacker does not know, here the attacker then need to get more data from the pairing devices. This extra information will be used to test if his brute force guess of the pass-key is correct. This data refers to the AU_RAND that the verifier sends to the claimant and expects the response ***SRES=*** E_2 ***(BD_ADDR***$_{claimant}$***, AU_RAND,*** K_{AB} ***)***. With this detail the attacker is now able to confirm his pass-key guesses. Thus if the pass-key is short the attacker has less work in guessing the it, thus short pass-keys are problematic in as far as protecting the pairing users from passive eavesdropper or man in the middle attack.

6.4 Key Disclosures

If a secret key (combination/unit etc) is disclosed to an attacker, then impersonation attack is feasible. However this has been solved by having the keys in a non readable form. The risk for disclosed keys is small for small personal devices e.g. headsets as their information content is often non-profitable. However more advance devices e.g. PDA's, mobile phones, laptops the risk is much higher as there are even more ways of coping the key if they are stored in readable forms. Various forms of attacks e.g. Trojan horses might even be used to get hold of this information. This is because if an attacker gets hold of the key it is possible for them to silently connect to device and use any of its services.

If an attacker is able to add a link key to a devices key database without proper pairing or change its label as trusted, this grants them unlimited access to all Bluetooth services running on the device. In reality this is complex to perform as it requires internal knowledge of the devices workings. An alternative to prevent this attack is for the device to incorporate some integrity check to the key database via extra parity bits.

A variation of the above approach is for the attacker to simply destroy the key databases of the device thus this error will be realized when an authentication is needed. As the specification does

not specify how to handle this situation, there are various approaches by manufactures. If the device implementation specifies that a new pairing be performed the attacker can then perform the passive pairing attack as described in subsection 4.3.

Unlike combination keys, a device that uses a unit key is only able to use one key for all its secure connections, thus it has to share this key with other devices that it trusts. Now supposing that an attacker is labeled as a trusted device in one instance communication with this device, it means that the device will be able to eavesdrop on the initial authentication messages of other two units that use the same unit key. This also gives the attacker impersonation capabilities. It is this unit key weakness, offering no protection against trusted attackers that lead to its eventual depreciation by the SIG.

6.5 Location tracking attacks

Location privacy is a real concern in Bluetooth enabled devices that benefit mostly from their anonymity in the access of various public services. Also since the devices that utilize the technology often contain private information e.g. mobile phones, PDA's etc. The threat is independent of whether the device is used for local connection or access technology as the radio signals from the device can always be tracked to a particular person. For the tracking to work there has to be a fixed device identity that the attacker tries link a person with, thus tracks their real life movements. This can be accomplished using the 48-bit ***BD_ADDR***.

In an inquiry attacks this mode of attack, the attacker distributes a number of Bluetooth devices in inquiry mode with a certain mesh, thus if a potential victim walks into this mesh with their devices in discoverable modes it becomes relatively easy for the attacker to precisely identify who they are and in sense link a face to the discovered ***BD_ADDR***.

6.5.1 Traffic monitoring attacks

In this type of attack the potential victim might have two or more devices not in discoverable modes but communicating thus not responding to inquiry request hence making the previous attack futile. However if the attacker decides to monitor all Bluetooth traffic whether it is meant for him or not then he will be able to detect the communicating devices and since the ***BD_ADDR*** is never encrypted he will be able to determine the communicating devices with more accuracy. The attack can further narrow down the devices by using IEEE's OUI.txt database to determine the devices manufacture and then query its SDP server for specific services.

A paging attack enables an attacker determine if a given device with a particular ***BD_ADDR*** is present within range. The attacker pages a particular ***BD_ADDR*** and if an ID packet is returned then the attacker know that the victim and device are within range. This work well as the victims device never report pages to upper layers that early. Tools exist that can brute force this mode of attack e.g. Redfang.

Frequency hopping attacks occurs when a Bluetooth device's hopping sequence is determined by various inputs e.g. $\boldsymbol{BD\_ADDR_{master}}$, master clock offset, etc. This information is not out of reach for an attacker thus theoretically proving that the attacker might be able to track and eavesdrop on the communication by hoping in the same random frequencies as the communicating devices.

The Bluetooth LMP name request command can be given an appealing name e.g. secret admirer, your friend, PIN 1234 etc. these names will often make the unsuspecting user to pair with the attacker device granting them total access to services. Thus their information is retrieved and enabling their identification. This type of attack is called a user-friendly attack.

6.6 Implementation flaws

Snarf attacks also known as bluesnarfing this attack takes advantage of a fault in Bluetooth implementation on certain mobile phones OBEX protocol is implemented. This silently access these mobile phones contacts, calendar and pictures without the owner ever knowing - a clear violation of the owner's security expectations. Some affected phone manufacturers are Nokia, and Sony Ericsson they have addressed the issues with updated firmware.

The backdoor attack is when a device that is no longer trusted can still gain access to the mobile phone and gain access to data as with bluesnarfing, as well as use services like WAP among others.

Bluebugging could possibly be the most publicized implementation attack. It allows the attacker to send/read SMS, call numbers, monitor phone calls and also do everything that backdoor and bluesnarfing allows. This is a separate vulnerability from bluesnarfing and does not affect all of the same phones as bluesnarfing

Bluejacking is an attack that capitalizes on the fact that during the initialization process, when a device wishes to be paired with you, a message containing the device's name and whether you want to pair with this device is displayed. To many people this is just an innocent joke to get a reaction out of someone by renaming their phone and then sending them a clever anonymous message and watching their reaction. Attacker can rename devices cleverly to enable pairing and thus gain access to the victim's device. It is very similar to user friendly name attack.

Mobile devices are also exposed to worms and viruses e.g. Cabir worm, which tries to pair the Bluetooth device it's on to any in the vicinity, and if successful it will install itself on the paired device. Once it is there, it will attempt to repeat this process, and also when the device is switched on, the worm will drain the battery by scanning for enabled Bluetooth devices.

The traditional DoS attacks may also be used on Bluetooth devices. The attacker keeps sending invalid Bluetooth requests that end up occupying a devices Bluetooth channel so it cannot communicate with any other Bluetooth devices.

7. Discussion

The SIG outlined in its roadmap through that it will continue to tackle issues of privacy and security. Nokia and Ericsson have both developed software upgrades for phones vulnerable to Bluetooth attacks. Both companies have also made sure that new phones coming to market will be able to defend against attacks.

Although users are advised to use long PIN codes, that minimize the risk of a security violation. Many scholars still regard Bluetooth as still being one of the most secure forms of wireless networking today. Bluetooth is considered more secure than other wireless technology, such as 802.11 networks and Wi-Fi which is vulnerable to security threats due to its weak WEP and WPA protocols. Everything considered Bluetooth is expected to become more pervasive as demand for wireless grows.

8. References

[1] www.bluetooth.com

[2] www.kjhole.com

[3] Hans Jacob Rivertz, “Bluetooth Security”, Nowergian Computing Centre, 3/3/2005.

[4] Gehrmann C., Joakim P. and Smeets B., "Bluetooth Security", 2004, artechhouse, inc, 685 Canton Street, Norwood.

[5] http://www.tech.plym.ac.uk/dcee/postgrad/reference/BlueTooth/page2.html

[6] T. Muller, “Bluetooth Security Architecture” White paper revision 1.0, Bluetooth Special Interest Group, July 1999

[7] http://www.cs.utk.edu/~dasgupta/bluetooth/bluesecurityarch.htm

[8] Bluetooth special interest group, specification of the Bluetooth system, version 1.2 core system package, November 2003

[9] http://www.eng.tau.ac.il/~yash/shaked-wool-mobisys05/

[10] www.embedded.com

[11] Jun-Zhao Sun, Douglas Howie, Antti Koivisto, and Jaakko Sauvola; Design, Implementation, and Evaluation of Bluetooth Security

[12]Dave Singelee,Bart Preneel,Security Overview of Bluetooth, COSIC Internal Report, June 2004

[13]Yaniv Shaked and Avishai Wool, Cracking the Bluetooth, PIN (http://www.eng.tau.ac.il/~yash/shaked-wool-mobisys05/)