Potential Bluetooth vulnerabilities in smartphones

Lih Wern Wong
Edith Cowan University

Potential Bluetooth Vulnerabilities in Smartphones

Lih Wern Wong
School of Computer and Information Science
Edith Cowan University
lihwern@yahoo.com

Abstract

Smartphone vendors have been increasingly integrating Bluetooth technology into their devices to increase accessible and convenience for users. As the current inclination of integrating PDA and telephony increase, the likelihood of sensitive information being stored on such a device is also increased. Potential Bluetooth vulnerabilities could provide alternative means to compromise Bluetooth-enable smartphones, leading to severe data breaches. This paper gives an insight on potential security vulnerabilities in Bluetooth-enabled smartphones and how these vulnerabilities may affect smartphone users. This paper is discussed from the viewpoint of Bluetooth weaknesses and implementation flaws, which includes pairing, weak key storage, key disclosure, key database modification, unit key weaknesses, manipulating sent data, locating tracking, implementation flaws, disclosure of undiscoverable devices, denial of service, device-based authentication, and uncontrolled propagation of Bluetooth waves, as well as Blueprinting and relay attacks.

Keywords:

Bluetooth Security, Bluetooth Vulnerabilities, Smartphones

INTRODUCTION

Increasingly, smartphones are used as a portable data storage and may potentially be used as a repository for account PINs, passwords and other private personal information. The current trend of integrating PDA and mobile phone technology into one device with Bluetooth add-on has increased the volume and spectrum of data that may be stored, relative to the device capabilities. Thus, security vulnerabilities on such devices may lead to more serious breach than would have been possible on a traditional low storage mobile telephony device. Though Bluetooth technology has all the advantages of wireless technology such as the ability to create a personal area network (a type of local area network in Bluetooth), it exhibits new and less publicized weaknesses.

Smartphone manufacturers usually produce their Bluetooth-enabled smartphones with lax security, which could compromise the security of the information held on such devices (Schwartz, 2004, p. 10). Bluetooth security is optional (Potter & Caswell, 2003). A common manufacturing default setting is to place smartphones in discoverable mode which removes authorization, authentication and encryption by default to increase the ease with which consumers may connect to accessories or other devices.

PAIRING

Bluetooth devices are susceptible to active and passive attacks during pairing procedure which usually occurs once between two devices. However such an attack is only possible, where the attacker is present during pairing procedure (Jakobsson & Wetzel, 2001, p. 10). Thus the threat posed by pairing is increased if such pairing is performed in public places such as a connection to an access point, laptop or vending machine. This section illustrates the attacks based on combination key, which easily apply to unit key as well. Combination key and unit key are two types of link keys that are used for authentication and link encryption key generation purposes between Bluetooth devices.

Pairing between devices A and B, as detailed in Figure 1, first requires the establishment of initialization key, K_{INIT}, which is calculated based on the above E_{22} algorithm, with input BD_ADDR (Bluetooth device address), IN_RAND (random number) and PKEY (secret passkey or PIN). K_{AB} is a combination key which is not transmitted individually through the air, but created using a local generated random value LK_RAND and it is exchanged via $K_{INIT} \oplus$ LK_RAND. Since both parties have a common understanding of K_{INT}, they are able to derive the respective LK_RAND.

K_{INIT} = E_{22}(BD_ADDR, IN_RAND, PKEY)
$K_{INIT} \oplus$ LK_RAND$_A$ (A send to B)
$K_{INIT} \oplus$ LK_RAND$_B$ (B send to A)
K_{AB} = E_{21}(LK_RAND$_A$, BD_ADDR$_A$) $\oplus$ E_{21}(LK_RAND$_B$, BD_ADDR$_B$)
SRES = E_1(BD_ADDR_claimant, AU_RAND_verifier, K_{AB})

Figure 1: Algorithms in Pairing Procedure

E_1 is used in authentication phase using K_{AB}, verifier AU_RAND and claimant BD_ADDR. During the pairing procedure between device A and B, the attacker is able to collect the following information in clear text over the air, as shown in Figure 2.

IN_RAND (random value)
BD_ADDR$_A$ (device A address)
BD_ADDR$_B$ (device B address)
$K_{INIT} \oplus$ LK_RAND$_A$ (random value)
$K_{INIT} \oplus$ LK_RAND$_B$ (random value)
AU_RAND (random value)
SRES (authentication output)

Figure 2: Data Transmitted in Clear Text

After such an attack, the only unknown parameter in the computation of current link key (combination key in this case) is the passkey. The attacker may then try a range of passkeys using the results to calculate the corresponding SRES, and later verify with the observed SRES value offline. Such an attack would be likely in two scenarios. First, attacker passively eavesdrops all the above communicated data between two pairing devices in clear. Then, attacker performs an offline brute force attack on a range of passkey and verifies the correctness of the passkey by matching the corresponding SRES' with the received SRES value, without sending any data to the victims.

Alternatively, attacker can actively engage in a pairing procedure by first faking the device address. After the establishment of link key using a guessed passkey, the attacker must first initiate the authentication procedure (perform sequentially) (Gehrmann, 2002, p. 7). Thus, the victim acting as the claimant will send the corresponding SRES to the attacker (verifier). After obtaining the SRES, the attacker uses brute force with a range of passkey values generated by calculating the corresponding initialization key. Next, for each initialization key, the attacker calculates the subsequent combination key, and then authenticate it locally using the obtained SRES (without interacting with victim) until a correct passkey is found.

The supposed 'security feature' in the Bluetooth authentication process of waiting certain amount of time to pass (which increases exponentially for each failed attempt) before a new authentication attempt in such a scenario would actually benefit the attacker by allowing progressively more time to compute the passkey on each attempt (Jakobsson & Wetzel, 2001, p. 11). Thus any smartphone using a short passkey for secure connection would be highly vulnerable to eavesdropping and man-in-the-middle attacks during the pairing procedure (Kugler, 2003, p. 9). The tendency of users to select short and easily identifiable passkeys serves to make this attack more feasible.

WEAK KEY STORAGE

As the Bluetooth specification does not specify how the Bluetooth host or controller should manage the link key database in term of access control and secure storage (Gehrmann, Persson & Smeets, 2004, p. 62), it is likely that threats may arise due to improper key storage handling.

Keys Disclosure

In most smartphone devices the key database is stored in non-volatile memory, thus making the acquisition of the database a non-trivial exercise. An attacker would require in-depth knowledge and special equipment to access the memory. However, in the case of smartphones which utilize external Bluetooth CompactFlash or SDIO (SD Input/Output) adapter, acquisition of link keys becomes a lot simpler though still requires physical access to the smartphones (Gehrmann, Persson & Smeets, 2004, p. 110). If the keys are stored in plain text in the smartphone memory, attacker could replace the actual Bluetooth adapter with a malicious adapter to retrieve the stored link keys using command HCI Link Key Request with BD_ADDR as parameter. However, using this method the attacker is only able to retrieve the link keys if there is a matching BD_ADDR.

Alternatively, if the key database is stored in the module (external card) instead of the host (smartphone), an adversary may simply remove the card and place it in another device and utilizing software which issues the HCI Read Stored Link Key command to read out all the link keys and their corresponding device addresses in the module (Bluetooth SIG, 2004, p. 480). Malware attacks are also possible using a Trojan horse which could disguise itself as a benign application and secretly retrieve the link key/address pairs and send it to adversary through various unprotected smartphone supported communication channels.

Once the key database is compromised, the attacker may use the link key and the associate device address (BD_ADDR) to effectively impersonate the actual device. For example, knowing the link keys and the computer Bluetooth adapter’s BD_ADDR the smartphone usually associates with, allows attacker to impersonate the computer and exploit the services the smartphone offers (e.g. make calls, SMS) over Bluetooth.

Key Database Modification

Unauthorized access to Bluetooth device without going through proper pairing can be granted by inserting or modifying link keys/address pairs in the key database. Thus rather than impersonating an existing pairing, an attacker could connect to a victimized smartphone using an inserted entry in the key database. If the link key is associated with a device marked as trusted, the attacker may use its now trusted BD_ADDR and obtain unrestrictive access to all Bluetooth services on the victimized smartphone (Karygiannis & Owens, 2002, p. 76).

This is further compounded by the Bluetooth specifications lack of procedures for the detection of a corrupted key database and lack of database integrity checking. Thus, if an attacker were to corrupt the keys database by altering all the link keys/address pairs, the device will then fail to authenticate with previously paired devices. This is further compounded as every time the devices fails to authenticate an increasing delay of service due to incremental waiting interval for repeat authentication attempts. To overcome this, the user will be required to reinitiate a new pairing and thus risk further attacks.

UNIT KEY

A unit key is usually used in devices that have limited memory availability such as Bluetooth mobile headsets, although Bluetooth 1.2 has withdrawn the use of unit key, the feature has been left in many devices to allow backwards compatibility with older Bluetooth devices (Bluetooth SIG, 2003, p. 79). However the use of a unit key as shared secret between two devices opens up more vulnerabilities in already insecure devices (Vainio, 2000). Unit keys are unique to each device and are usually never changed as devices utilizing a unit key are only able to use one shared key for all secure connection with other trusted devices.

An example would be the communication between device A (e.g. victim’s smartphone) and B (e.g. headset) which communicate using A’s unit key as their link key. If device C (e.g. attacker’s laptop) has previously communicated with device A, it will already possess the unit key which may then be utilized to eavesdrop on communication between devices A and B. Attackers might also use such knowledge to impersonate device B, or at worst perform man-in-the-middle attack. Consequently, unit key is unable to protect device issuing the unit key against attack from any trusted device.

MANIPULATING SENT DATA

An attacker may send erroneous data to the receiver by inserting or modifying the data originally sent. In situations where encryption is not present, attackers may easily modify the actual user data and its corresponding CRC (Cyclic Redundancy Check) in the packet’s payload field to create meaningful and effective changes that might go unnoticed by receiver (BSI, 2003, p. 7). While posing greater difficulty, encrypted data may still be modified through the use of a newly computed CRC value which agrees with modified data, as the encryption and CRC computation are linear operations (Gehrmann, Persson & Smeets, 2004, p. 105). However such manipulation on encrypted payloads will only disrupt the Bluetooth communication, without actually allowing significant changes on end-user data.

LOCATION TRACKING

With the integration of Bluetooth technology into personal mobile devices which are carried around by their owners such as smartphones, PDA’s or laptops, the location privacy of users comes under threat. A Bluetooth-enabled device carried and used by a person may be tracked using the radio signal transmitted from the device. Once the attacker is able to associate the device identity (e.g. unique Bluetooth device address or non-unique device friendly name) with the actual user identity, user movement may be tracked The long promised anonymity mode (Kewney, 2003) which was introduced to mitigate such attack by having changeable device address (Graham, n.d.), was not mature enough to be included in latest Bluetooth 2.0 specification.

Inquiry Attack

Bluetooth device address (BD_ADDR) is best used to uniquely identify a Bluetooth-enabled device. Attacker could exploit existing public access Bluetooth infrastructure or deployed numerous Bluetooth devices in a targeted area to pinpoint users’ location. Suppose victim device is set to discoverable mode, attacker can frequently perform inquiry scan to discover Bluetooth-enabled device and log all device addresses that are discovered with time associated. The logs would reveal users movement and possible their association (i.e. relation exist between two people who frequently gather at the same area) (Gehrmann, Persson & Smeets, 2004, p. 115).

Traffic Monitoring Attack

Even devices in non-discoverable mode may be located through the monitoring of communication between devices. All packets in Bluetooth traffic has an access code field which value is derived from the master device address in the piconet, called channel access code (CAC). Receivers use CAC to identify packets that belong to the corresponding piconet. Thus, by monitoring traffic in an area, attacker can determine the existence of master devices. Though CAC is not unique, the existence of two devices having the same CAC in a small area is rare (Gehrmann, Persson & Smeets, 2004, p. 115). While FHS (frequency hopping sequence) packets or packets with DAC (device access code) value can be monitored to also determine an individual device, such methods are not as effective as CAC as they are only used during connection establishment.

Paging Attack

When a device is in connectable mode, attacker could page a particular BD_ADDR or DAC to determine if the device is within range, without responding back (Jakobsson & Wetzel, 2001, p. 12). The paged device is found if the device response to the paging request. The victimized device will then timeout without alerting the user that no response was received.

Frequency Hopping Attack

The Bluetooth frequency hopping scheme is a repeating frequency hopping sequence, determined by master device address and clock. In paged state, the LAP (lower address part, 24-bit) and four LSB (least significant bit) in UAP (upper address part) of the paged device BD_ADDR (48-bit) is used. In connection state, the LAP and four LSB in UAP of master device (i.e. paging device) BD_ADDR is used. Theoretically, attacker could monitor the frequency hopping scheme and determine the LAP and four LSB in UAP (Gehrmann, Persson & Smeets, 2004, p. 116). Thus, partial (28-bit out of 48-bit) of the master device BD_ADDR can be acquired throughout the connection state, while acquiring partial of the slave device BD_ADDR during paging state.

User-friendly Name Attack

After the completion of paging procedure, an attacker could request nearby devices for their user-friendly name to locate the victim device, assuming that a fairly unique user-friendly name is defined on the victim device. This attack requires the victim device to be in a connectable state (i.e. could be paged).

IMPLEMENTATION FLAWS

Some aspects of Bluetooth security, such as security policy enforcement, key database management and others, which contribute to the overall device security, are not assessed vigorously by smartphone vendors as they are not mandated in the current Bluetooth specification. Such oversight might potentially lead to defective implementations, such as the following Bluetooth vulnerabilities.

BlueSnarf

BlueSnarf allows an adversary to connect to a Bluetooth-enabled device without going through normal pairing, and subsequently access stored data such as phonebook information (e.g. data and image related with an individual entry), calendar details, real-time clock settings, stored business cards and IMEI, (International Mobile Equipment Identity) without alerting the owner (Laurie & Laurie, 2004). Actual field assessment in CeBIT 2004 confirms such BlueSnarf attack (Herfurt, 2004). This attack is feasible on flawed devices regardless of discoverable mode.

In order to perform BlueSnarf attack, the attacker first needs to connect to the OBEX Push Profile (OPP). OBEX Push is a service that usually does not require authentication (though OBEX is capable of managing authentication) to ensure proper implementation and it is commonly used for business card exchange or uploading and retrieving other similar objects. Attacker could possible exploit such vulnerability via IrDA or serial port as this attack or vulnerability is unrelated to OBEX protocol (Rowe & Hurman, 2003).

OBEX vulnerabilities can be approached through GET and PUT, which refer to object retrieval and uploading respectively. Attacker connects to the OBEX Push recipient and issues an OBEX GET request to potential vulnerable smartphone to retrieve all files with known or correctly guessed filename, such as 'telecom/pb.vcf' for the devices phonebook or 'telecom/cal.vcs' for the devices calendar file (refer IrMC Specification for standard filenames) (BlueSnarf, 2004). Usually, such files are only accessible through protected profile. However, improper implementation causes them to be accessible via unprotected profile such as OBEX Push. Alternatively, attack could push a PUT request with empty body, which refers to DELETE action in OBEX. By pushing the request to target's phonebook entries, it will potential overwrite or remove the phonebook entries (Rowe & Hurman, 2003).

A slight variation of BlueSnarf called BlueSnarf++ exists, which allows attacker to have full read/write access to vulnerable device file system by connecting to the OBEX Push Profile. Instead of the simple OBEX Push function, these flawed smartphones' allows attacker to connect to the OBEX File Transfer Profile as OBEX Push without pairing (BlueSnarf++, 2004). This allows attacker to view and manipulate the files in the file system including external memory sources, such as SD cards and memory sticks.

Backdoor Attack

Flawed implementation allows some smartphones to leave credential information on a device even though the previous pairing information has been removed from the victim's list of paired device (Laurie & Laurie, 2004). Full access to the device is still granted according to the previous trust relationship without the owner's knowledge as the pairing entry is invisible to the user. An attacker could secretly pair the devices, and deliberately delete the pairing entry, while still having covert access to the victimized device. Aside from retrieving or manipulating smartphone data, potential services attacker could access without victim consent include modem, WAP and GPRS services. Unless user is observing their smartphone and notices the small icon indicating an established Bluetooth connection, the user would be unaware of the unauthorized access.

BlueBug

BlueBug is a device security flaw that allows an attacker to issue AT (ASCII Terminal) commands to vulnerable smartphone via Bluetooth without alerting the user. AT command set is commonly used to configure and control communication devices like modems. If successful, the adversary will have application level control over the smartphone. An adversary could exploit AT commands to make and divert calls, manipulate phonebook, update calendar, send/read/delete SMS, connect to Internet, configure phone settings and so forth (BlueBug, 2004).

For instance, call forwarding allows attacker to intercept or impersonate the victim by diverting all victim incoming calls to attacker. Some service providers allow user to track their customers' location through GSM cell networks, after seeking permission from the customers, which is usually obtained via SMS, as an attacker controls the victim's phone and they may approve the request without the users' knowledge (BlueBug, 2004).

Attacker could use the smartphone as bug to eavesdrop on conversations which take place around the phone by instructing the smartphone to make a call over GSM network to the attacker (e.g. using anonymous prepaid phone number) which may situated anywhere. Furthermore, victimized smartphones could be instructed to call a premium number service which the attacker has affiliation with, causing serious financial lost to the victims.

Various vendors implement Bluetooth stack poorly, they create some unpublished service in RFCOMM channel, which is not available in Service Discovery Profile (SDP). Attacker can connect to unpublished headset service via RFCOMM without pairing (Laurie, Holtmann & Herfurt, 2005).

Bluejacking

Bluejacking allows users to send anonymous business card or phonebook contact to Bluetooth-enabled smartphone in discoverable mode. It is achieve using OBEX protocol object push feature. Attacker creates a phonebook contact with flirtatious message in the "name" field and broadcast it in crowded area. Fortunately, this attack does not modify or delete any smartphone stored data, but annoys or harasses user with unsolicited flirtatious message rather then usual name and phone number business cards they would expect. A Bluejacker usually send more personal messages if the victim responses. Vendors can use such vulnerability as a marketing tool to post advertisement. However, an attacker could possible send business card which may overwrite the existing phonebook entry having the same name to unaware victims, hoping they would blindly accept the entry, for example replacing “home” entry with premium rate number (Whitehouse, 2004).

Another variation of Bluejacking exploits the pairing procedure in which Bluetooth devices establish a secure connection. This attack is feasible as the “name” of the attacker device initiating the pairing is display to the target device as part of the pairing procedure, and the name field allows text up to 248 characters, enough for the attacker to construct a meaningful message (Laurie & Laurie, 2004). Most users would probably be careful enough not to complete the pairing with unknown devices. While such Bluejacking may seem rather benign, and not much of a real security threat aside from causing annoyance, attacker could deceive unwitting user by constructing tempting message. For instance, message like “You have won $ 20,000 in XXX draw! Enter this 4 digits PIN and call 09-64323456 to collect the prize!” would likely be successful in deceiving targets to complete the pairing without raising much suspicion. Consequently, if the pairing is successful, services or data on the device will be accessible to the attacker.

BlueSmack

BlueSmack is a Bluetooth version of “Ping of Death” (BlueSmack, 2004). It utilized L2CAP layer’s L2CAP ping (echo request) which is used to check connectivity and connection roundtrip time. This attack attempts to overwhelm the vulnerable device by sending large ping packets without requiring an open L2CAP channel, which could possibly cause buffer overflow or denial of service attacks (Laurie, Holtmann & Herfurt, 2005). Since L2CAP signal MTU is unspecified, attacker can customize the ping packets size (Bluetooth SIG, 2004, p. 57).

DISCOVERY OF UNDISCOVERABLE DEVICES

Even device in non-discoverable mode can be located using tools like @Stake’s RedFang. RedFang operating in multithread environment, utilizing up to 8 Bluetooth adapters could reduce the 11 hours in single threat computation to approximately 90 minutes (Whitehouse, 2003, p. 8). Since a device must align to the correct hopping pattern to be able to communicate with a BD_ADDR, the operation takes considerable amount of time. Though non-discoverable device does not broadcast it present (i.e. does not respond to inquiry scan request), it does response to certain request. Devices in non-discoverable mode response to direct name and services inquiry request, as non-discoverable authenticated or bonded devices require some sort of mechanism to update services and name (Whitehouse, 2003, p. 6). Attacker will have to brute force through the device BD_ADDR to make name and services inquiry request to discover invisible devices. Only the 24-bit LSB of BD_ADDR is gone through, while keeping the 24-bit MSB of BD_ADDR fixed to a particular manufacturer, since it is unique for each Bluetooth interface manufacturer (Potter & Caswell, 2003).

DENIAL OF SERVICE (DoS)

As previously mentioned Bluetooth-enabled smartphones are particularly susceptible to the DoS attacks. Constant inquiring or requesting response from the receiving smartphone will degrade its battery (Singel´ee & Preneel, 2004). After an attack, a user may be unable to use their smartphone since the attack has drained off the battery completely.

Another possible scenario for a DoS would be to overwhelm the smartphone with invalid Bluetooth packets by the clogging communication channels, the owner could temporarily be refused of other Bluetooth services. Bluetooth physical communication link are used by two logical channels, SCO (Synchronous Connection-Oriented) and ACL (Asynchronous Connection-Oriented). Additionally, Bluetooth device supports a maximum number of simultaneous active connections with 2Mbps bandwidth in current Bluetooth 2.0. To clog up the bandwidth, attacker could impersonate a trusted device and paired with the victimized smartphone and make request without acknowledgement for the recipient of packets. The smartphone will retransmit asynchronous data that transport over ACL link once the acknowledgement of recipient is not received. Thus, using multiple devices and requesting large amount of asynchronous data, attacker could potentially deny the smartphone from serving other devices, since the ACL link is constantly retransmitting data. In the case of maximum 7 active connections, attacker could impersonate all 7 slave devices and occupy the victimized smartphone (master) with bogus traffics. Consequently, other devices that wish to connect to the victimized smartphone will be forced to switch to parked state while still synchronize with the master (Bluetooth SIG, 2004, p. 45). Thus, attacker could consume all the bandwidth and maximum active connections to cause DoS (Niem, 2002, p. 17)

Since there is a waiting interval after each failed authentication before a new attempt can be made, attack can exploit it to cause denial of service. Bluetooth-enabled device (verifier) kept a list of devices that failed the authentication procedure, which is defined by the claimant BD_ADDR. Attacker could impersonate a target device (claimant) DB_ADDR by changing attacker device DB_ADDR and send erroneous authentication response to verifier. The verifier will disregard the SRES value (though correct) sent by the victimized claimant after receiving the erroneous SRES sent by the attacker (assuming attacker sent its SRES faster than the victim’s SRES), and fail the authentication (Candolin, 2000). Thus, the victimized device would be prevented from authenticating itself to the verifier.

DEVICE-BASED AUTHENTICATION

Bluetooth security architecture only provides device-based authentication without user-based access control (Karygiannis & Owens, 2002). In most cases, user does not have to authenticate himself to the Bluetooth module (user only need to authenticate himself to the smartphone if user-based access control is activated). Basically, attacker will just have to impersonate target smartphone’s DB_ADDR by changing attacker own’s BD_ADDR to gain access to devices the target smartphone has previously paired with. It could cause problem to smartphone that support external Bluetooth adapter. Attack could simply steals the adapter and use it on other devices to gain access to devices the adapter has previously paired with, without initiating a pairing (assuming link key/address pairs information is stored in the adapter).

UNCONTROLLED PROPAGATION OF BLUETOOTH WAVE

Use of high-gain directional antenna or standard 2.4GHz 802.11 and amplifier to boost the weak Bluetooth signal allows eavesdropping from a distance greater than the usual range (Cheung, 2005). If the traffic is not encrypted, attacker will be able to learn the transmitted user data from far without raising any suspicion. Bluetooth transmission power control is optional and most smartphones do not support such feature. Thus, the usual short Bluetooth range transmitted by most smartphones (Class 2 device with up to 10 meters) does not actually protect users from eavesdropping. Besides, using high-gain antenna, attacker will have more time (as users is still within the receiving and transmitting range after moving more than 10 meters) to perform attacks at a much safer distance.

For instance, from far, attacker could transmit low-level jamming signal to disrupt the connection, causing a reconnection between victim’s smartphone and external keyboard. By eavesdropping the pairing procedure, attacker could possible calculate the passkey and subsequently record every keystroke (Schwartz, 2004, p. 10).

BLUEPRINTING

Every Bluetooth-enabled device has unique attribute (e.g. device address), including manufacture specific (e.g. first 24-bit MSB of BD_ADDR) and model specific attributes (service description records). Blueprinting is a technique to obtain such device information to determine the device manufacturer, model type, and even firmware version on some smartphones. Blueprinting is Bluetooth version fingerprinting. Attacker can use Blueprinting to uncover potential vulnerable smartphones.

Bluetooth-enabled device address is a 48-bit identifier similar to MAC address (e.g. MM:MM:MM:NN:NN:NN). The first half of the address refers to Bluetooth chipset manufacturer, which usually identifies the device manufacturer (IEEE, 2005). However, the last 24-bit is randomly assigned by manufacturer, which give no notation of device model. Fortunately, Service Discovery Protocol (SDP) profile can be used to refine the identification. Remote devices query SDP to enquire services offered by a Bluetooth-enabled device, and how to utilize the respective services. Thus, by hashing certain values in the SDP Profile response and calculating a fingerprint value, attacker can associate the value to respective device model (Blueprinting, 2004).

According to the Bluetooth specification, all Bluetooth devices has Service Record Handle field, a 32-bit number that uniquely identifier each service with SDP server during device startup. The Record Handle value in a smartphone is hard coded within firmware. RFCOMM channel or L2CAP PSM number that the service can be accessed under is also included in computing the hash. Part of the Blueprinting hash includes the sum of Record Handle value multiplied by channel for all running services on the smartphone (e.g. (RecHandle1*Channel1) + (RecHandle2*Channel2) +...+ (RecHandlen*Channeln)) (Herfurt & Mulliner, 2004). Thus, Blueprinting uses combination of manufacture part of BD_ADDR and actual fingerprint (i.e. hash value) as identifier (e.g. 00:60:57@2621543 , fingerprint for Nokia 6310).

RELAY ATTACK

In relay attack, attacker C communicates with A impersonating as B, and communicate with B impersonating as A. Relay attack is quite similar to man-in-the-middle attack, however, attacker does not need to know the passkey or link key. Attacker basically relays message between A and B without modifying the contents. Relay attack is only possible if victims A and B employ only device-based authentication without application level authentication or encryption (Levi, Cetintas, Aydos, Koc, & Ufuk, 2004, p. 10). Attacker C could switch between the role of AC and BC by impersonating respective devices BD_ADDR.

C must wait for actual connection request from either A or B. Suppose A wants to connect to B by first paging BC, thinking it was B, as shown in Figure 3. Meanwhile, AC pages B to complete the paging procedure. A sends LMP_host_connection_req command, which is relay by BC to B. AC accept the connection request response from B and relay it to A through BC. In this connection establishment, C must respond faster to the A’s paging request than B, so that A will connect to C instead of B (Kugler, 2003, p. 3). AC must use a different frequency hopping sequence from A so that A and B cannot discover each other (Device that initiates paging is the master, who determine piconet frequency hopping sequence).

The changing of link keys does not affect the attack. C basically has to relay the encrypted random value (RAND_NR) between A and B to complete the calculation of combination key. In authentication phase, C relays the challenge (AU_RAND) and response (SRES) between A and B to complete the authentication. Thus, A thinks B is authenticated, but it is actually BC that is authenticated. Similarly, B thinks A is authentication, which is actually AC. Thus, since C is able to relay traffic between A and B, it could possible cause DoS by delaying or stop the relaying process.

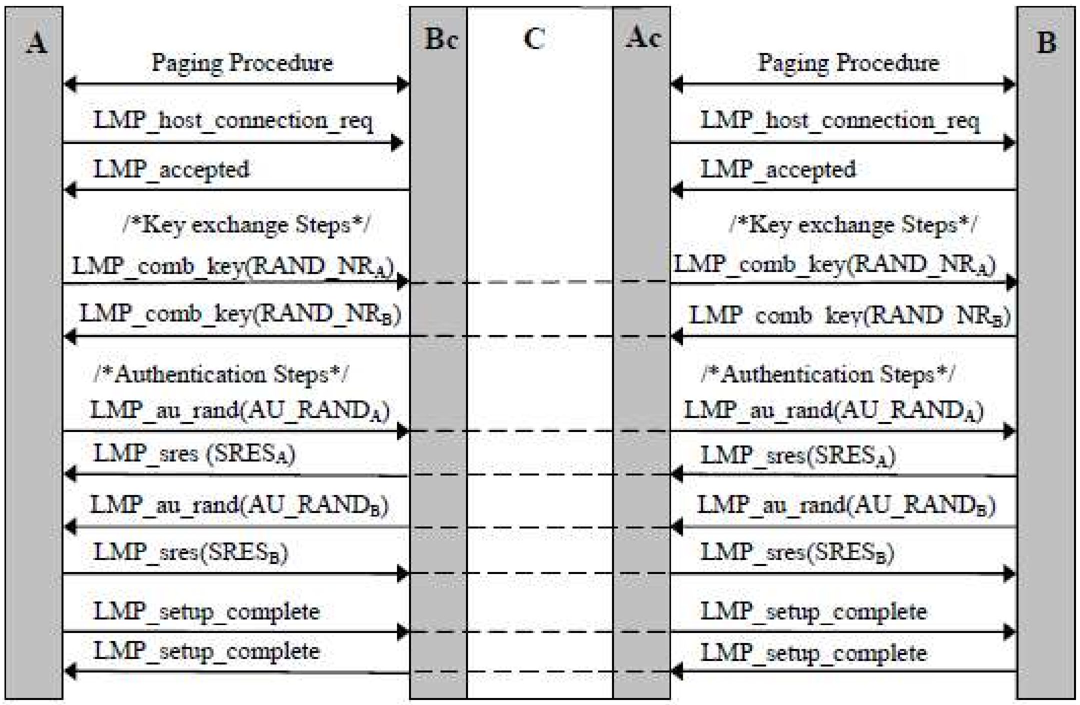

Figure 3: Relay Attack (Levi, Cetintas, Aydos, Koc, & Ufuk, 2004, p. 4)

CONCLUSION

Most Bluetooth-enabled smartphones has default lax security settings to increase the ease of which consumers can connect to accessories or devices. Moreover, most users are unaware of the potential risk associated with such devices or are not knowledgeable on the customization of Bluetooth settings. Such factors have caused some of the potential vulnerabilities or attacks discussed above very viable on smartphones. Potential abuses in smartphones will increase with the introduction of Wi-Fi technology in smartphones, as Wi-Fi offers greater wireless range and bandwidth when compared to Bluetooth. Smartphones users need to be made aware of the potential risk associated with such devices. As the adoption of Bluetooth-enabled smartphones increases, so will the potential risk. While Bluetooth technology does introduce some threats to users devices, users should always practice some basic safety procedures when using such devices which will significantly mitigate the potential risk. Such procedures include switching off Bluetooth functionality whenever not in use, employ authentication and encryption whenever possible, and never pair with unknown devices. Thus, until Bluetooth SIG introduces and enforces more robust and secure security mechanism and smartphone manufacturers produce more secure smartphones, smartphone users should always use Bluetooth communication with caution.

REFERENCES

BlueBug (2004). Retrieved May 13, 2005, from http://trifinite.org/trifinite_stuff_bluebug.html

Blueprinting (2004). Retrieved May 13, 2005, from http://trifinite.org/trifinite_stuff_blueprinting.html

BlueSmack (2004). Retrieved May 13, 2005, from http://trifinite.org/trifinite_stuff_bluesmack.html

BlueSnarf (2004). Retrieved May 13, 2005, from http://trifinite.org/trifinite_stuff_bluesnarf.html

BlueSnarf++ (2004). Retrieved May 13, 2005, from http://trifinite.org/trifinite_stuff_bluesnarfpp.html

Bluetooth SIG (2003, November 5). Specification of the Bluetooth System, Version 1.2. Retrieved May 13, 2005, from https://www.bluetooth.org/foundry/adopters/ document/Bluetooth_Core_Specification_v1.2

Bluetooth SIG (2004, November 4). Specification of the Bluetooth System, Version 2.0 + EDR. Retrieved May 13, 2005, from https://www.bluetooth.org/foundry/adopters/ document/Core_v2.0_EDR/en/1/Core_v2.0_EDR.zip

BSI (2003). Bluetooth - Threats and Security Measures. Retrieved May 13, 2005, from http://www.bsi.bund.de/english/publications/brosch/B05_bluetooth.pdf

Candolin, C. (2000, October 23) Security Issues for Wearable Computing and Bluetooth Technology. Retrieved May 18, 2005, from http://www.tml.hut.fi/~candolin/Publications/BT/btwearable.pdf

Cheung, H. (2005, March 8). How To: Building a BlueSniper Rifle – Part 1. Retrieved May 16, 2005, from http://www.tomsnetworking.com/Sections-print-article106.php

Gehrmann, C. (2002, April 19). Bluetooth™ Security White Paper. Retrieved May 12, 2005, from http://grouper.ieee.org/groups/1451/5/Comparison%20of%20PHY/ Bluetooth_24Security_Paper.pdf

Gehrmann, C., Persson, J., & Smeets, B. (2004). Bluetooth Security. Massachusetts, United State of America: Artech House.

Graham, R. (n.d.). Bluetooth Security Mechanisms. Retrieved May 16, 2005, from http://www.rfi-global.com/pdfs/Bluetooth_Security_Article.pdf

Herfurt, M. (2004, March 30). Bluesnarfing @ CeBIT 2004. Retrieved May 14, 2005, from http://trifinite.org/Downloads/BlueSnarf_CeBIT2004.pdf

Herfurt, M., & Mulliner, C. (2004, December 20). Blueprinting - Remote Device Identification Based on Bluetooth Fingerprinting Techniques. Retrieved May 13, 2005, from http://trifinite.org/Downloads/Blueprinting.pdf

IEEE (2005, May 16). IEEE OUI and Company_ID Assignments. Retrieved May 16, 2005, from http://standards.ieee.org/regauth/oui/oui.txt

Jakobsson, M., & Wetzel, S. (2001). Security Weaknesses in Bluetooth. Retrieved May 14, 2005, from http://www.informatics.indiana.edu/markus/papers/bluetooth.pdf

Karygiannis, T., & Owens, L. (2002, November). Wireless Network Security 802.11, Bluetooth and Handheld Devices. Retrieved May 10, 2005, from NIST website: http://csrc.nist.gov/publications/ nistpubs/800-48/NIST_SP_800-48.pdf

Kewney, G. (2003, November 3). Bluetooth shock delay to version 1.2 -"hopefully this year". Retrieved May 16, 2005, from http://www.newswireless.net/index.cfm/article/864

Kugler, D. (2003). Man-in-the-Middle Attacks on Bluetooth. Retrieved May 16, 2005, from http://www.cs.stevens.edu/~swetzel/CS693/spring05/papers/wireless/kuegler.pdf

Laurie, A., Holtmann, M., & Herfurt, M. (2005, March). Hacking Bluetooth Enabled Mobile Phones and Beyond – Full Disclosure. Retrieved May 15, 2005, from http://opensores.thebunker.net/pub/mirrors/blackhat/presentations/bh-europe-05/BH_EU_05-Laurie_Herfurt_Holtmann/BH_EU_05_Laurie_Herfurt_Holtmann.pdf

Laurie, A., & Laurie, B. (2004, October 14). Serious Flaws in Bluetooth Security Lead to Disclosure of Personal Data. Retrieved May 11, 2005, from http://www.thebunker.net/security/bluetooth.htm

Levi, A., Cetintas, E., Aydos, M., Koc, C.K., & Ufuk, M. (2004). Relay Attacks on Bluetooth Authentication and Solutions. Retrieved May 16, 2005, from http://islab.oregonstate.edu/koc/papers/c32relay.pdf

Niem, T.C. (2002, November 4). Bluetooth and Its Inherent Security Issues. Retrieved May 16, 2005, from http://www.sans.org/rr/whitepapers/wireless/945.php

Potter, B., & Caswell, B. (2003). Bluesniff – The Next Wardriving Frontier. Retrieved May 16, 2005, from http://www.shmoo.com/~gdead/dc-11-brucepotter.ppt

Rowe, M. & Hurman, T. (2003, November 13). Re: Serious flaws in Bluetooth Security Lead to Disclosure of Personal Data. Retrieved May 15, 2005, from http://lists.virus.org/bugtraq-0311/msg00165.html

Schwartz, E. (2004, July 26). Beware, Bluetooth Can Bite. InfoWorld, 26(30), 10. Retrieved May 14, 2005, from http://0-proquest.umi.com.library.ecu.edu.au/pqdweb?did=679695161&sid=2&Fmt=4&clientId=7582&RQT=309&VName=PQD

Singel´ee, D., & Preneel, B. (2004, June). Security Overview of Bluetooth. Retrieved May 15, 2005, from http://www.cosic.esat.kuleuven.be/publications/article-565.pdf

Vainio, J.T. (2000, May 25). Bluetooth Security. Retrieved May 13, 2005, from http://www.niksula.cs.hut.fi/~jiitv/bluesec.html

Whitehouse, O. (2003, October). War Nibbling: Bluetooth Insecurity. Retrieved May 14, 2005, from http://www.atstake.com/research/reports/acrobat/atstake_war_nibbling.pdf

Whitehouse, O. (2004, July). Bluetooth Still Needs Security Bite: Using Bluetooth can be a Risky Business, Particularly for Those in Business. Retrieved May 16, 2005, from http://www.findarticles.com/p/articles/mi_m0IUL/is_7_38/ai_n6175563/print

COPYRIGHT