I.J. Information Engineering and Electronic Business, **2015, 4, 58-75**
Published Online July 2015 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijieeb.2015.04.08

An Augmented Level of Security for Bluetooth Devices Controlled by Smart Phones and Ubiquitous Handheld Gadgets

Soham Sengupta[1]
Faculty of Information Technology, JIS College of Engineering, &
Research Scholar, DETS, University of Kalyani, India
Email: soham.sengupta.java@gmail.com

Dr. Partha pratim Sarkar[2]
DETS, University of Kalyani, India
Email: parthabe91@yahoo.co.in

Abstract—The enormous growth of smartphones was impelled by the idea to make a mobile phone offer more than just cellular telephony. One of the prime factors that initiated the age of smartphones (e.g. *iOS*, *Android*, *RIM*, etc.) was inarguably the capability of wireless sharing of images, music etc. among the users; which was possible due to Bluetooth Technology (IEEE 802.15). Today customers of the cheapest phone in world demand to have an inbuilt Bluetooth stack. Apart from sharing files, especially media, Bluetooth provides us with a lot more functionality, like streaming audio to a home entertainment system, allowing to share an Internet connection over DUN profile, a remote car locking and security system, a few to mention. Though the *IEEE 802.15* stack has its own security mechanism, sometimes a system might require an additional security architecture running collaboratively with the in-built security to authorize an inbound pairing request. A simple example of the authorization paradox is that the standard security mechanism cannot help a Bluetooth system that was paired to multiple devices, to decide which of the paired devices to authorize to execute a certain task. For example, a device may be required to allow a smartphone Bluetooth stack to stream audio but restrict it from transferring files. Here need of a profile specific authorization is felt but it is beyond the scope of IEEE 802.15.

To understand it better, let us assume that a home theater system has a Bluetooth link which allows smart phones to stream audio to it over A2D Audio sharing profile. Such a home theater system (e.g. HT-DZ350 by Sony) can be connected to any smartphone and play the streamed music. Each time a device disconnects, the Bluetooth stack resets itself and identity of the Bluetooth stack on the smartphone is lost. Since Bluetooth radio waves can penetrate walls and windows, it may be possible that a neighbor of mine connected her smartphone to the Home theater system and played an unwanted music. Sometimes this can be fatal in some remote controlled instruments unless proper security mechanisms are installed.

Proposed in this thesis is a novel, generic and extensible framework to prevent unauthorized access over Bluetooth serial port profile; which is independent of any Cryptographic algorithm or approach. The thesis also suggests different architectures for differently equipped hardware systems, because the performance of the system under an augmented security stack will be different for different devices with varying hardware resources.

Index Terms—Bluetooth Security Model, Bluetooth Serial port Profile, IEEE 802.15 DUN Profile, Proprietary Security Mechanism, JSR-82.

I. Introduction

Though Bluetooth technology has got wide acceptance and popularity as a wireless link for file transfers and entertainment of users, it was envisioned by the **SIG** to achieve bigger causes including remote control of devices, Dial up networking (DUN), audio streaming (found in a Bluetooth headset and hands-free) and many other services provided by different profiles of IEEE 802.15. While use of Bluetooth is found everywhere, from a feature phone to home entertainment systems, domestic electrical appliances, car locking system, and home security solutions; we observe that Bluetooth Serial Port Profile (***btspp***) is the fittest among all its available profiles to achieve a cross device connectivity. This very profile of Bluetooth allows different applications running on smartphones to connect to a proprietary Bluetooth stack and execute remote control operations. As with every other remote control device, the need of security becomes indispensable with Bluetooth serial Port profile, as well. The security mainly focuses on the identity of the controller Bluetooth stack which gets access to a Bluetooth device. In brief, authentication and authorization become the most priority factors in certain situations. To understand it better, let us assume that a home theater system has a Bluetooth link which allows smart phones to stream audio to it over Audio-Sharing profile. Such a home theater system (e.g. HT-DZ350 by Sony Electronics) can be connected to any smartphone and play the streamed music. Each time a device disconnects, the Bluetooth stack resets itself and identity of the Bluetooth stack on the smartphone is lost. Since Bluetooth radio waves can penetrate walls and windows, it may be possible that a neighbor of mine connected her smartphone to the Home theater system and played an unwanted music. Similar situation arises when an Electrical device which can be turned off or on using a smartphone application, is accidentally turned off or on by my niece who was playing with my Tablet where the same controller application was installed.

The proposed model addresses the difference of degree of security needed by a variety of devices equipped with different hardware specification. This promises to fit in a very low-end MCU where complex cryptographic algorithms cannot gracefully execute and make the system too sluggish to use. The thesis also presents a modified metaphor of the architecture to fit for a costly hardware, where need of security is highly prioritized and the system is capable of executing complex algorithms of Asymmetric cryptographic approach. Besides proposing the model, the thesis presents a comprehensive analysis of system responsiveness using both symmetric and asymmetric Cryptography and shows how this model serves existing solutions in an efficient and extensible approach

II. Proposed Bluetooth Security Model

The proposed model aims to serve for 3 broader categories of devices and systems, which are:

a) Small MCU driven devices having low computational resources and requiring normal domestic security of commonplace users; i.e. people whose activities are not being monitored and who are innocent politically, professionally and personally.
b) Small MCU driven systems having little to moderate computational resources and to be used by innocent people, whose activities are not being monitored or who are not vulnerable to professional, political or other intentional attacks. But these devices are used to perform some everyday task, which cannot be allowed to malfunction for safety reason.
c) High end computing systems which are costly and hence used in sectors where probability of an attack for device malfunction or delayed-function is very high.

2.1 Prior Art and Case Studies

The first category of devices can be exemplified as a low cost power supply controller for less important household appliances like the Music System, Television, Lamps or FAN where a malfunction of the power system does not have a serious effect on the household or to individuals who use this type of devices. If one wished to turn off or on any of these appliances and the device malfunctioned or delayed to function, there would have been no serious consequence. Also, coming to the issue of security of such devices operating over ***Bluetooth Serial Port Profile***, we readily conclude the following:

a) These devices are low in cost and have little computational power. Hence they are supposed to be used by sectors or individuals who never need highly invincible security, since high end Cryptography algorithms will either fail on low computational resources or make the system too sluggish.
b) The only intention behind a possible attack may be simply silly and innocent practical jokes or pranks by kids.
c) Even if an attack is made and that succeeds, it would not have a serious consequence. For example, a Kid turning off the television using his mother's smart phone, while his Father is watching the News Updates on it, shall not have much of a serious consequence causing some significant damage.

For such systems, one should be satisfied with simple Cryptographic algorithm, if he cannot accept a no-security or open-to-access device. These devices, therefore, can be modelled to provide some basic security, as generalized in **[Table.1]**.

This type of devices, however, may be used in domestic households. Certain domestic appliances, like an Air Conditioner, which do not involve extra vulnerability other than kidding or pranks as described above; may sometime fall prey to a kid's pranking attack to turn it on in absence of his parents, leading to three major unwanted consequences:

- ✓ Unnecessary consumption of electricity implying unwanted Electricity Bills
- ✓ The Air Conditioning machine kept turned on for a long time with the doors and windows open, might damage the AC itself.
- ✓ In a household where a 15 Ampere cabling connects to two Air Conditioners, which are not supposed to operate simultaneously, a prank by a kid to turn an AC on while another is already running, might cause fire caused by excessive heating of the connector box; if no proper measures like circuit breakers were installed. Such a situation is very rare in countries in Europe, US, Japan and other parts of the first world. But in countries belonging to the third world, like portions of India, two ACs sharing same connector box, without proper circuit-breakers installed, is very common.
- ✓ Further, consider a manufacturing plant, where the workers control electro-mechanical machines with their smart phone applications, such pranks or

careless use of Bluetooth serial port profile might have dire consequences, if a worker connects over the wrong Bluetooth Serial Port Profile maybe, unintentionally, and turns on, let us say, a fiber reeling machine all off a sudden, where maintenance was in progress. The person(s) who had been working on maintenance of this machine, would lose their limbs, if they at all manage to survive the accident. In order to prevent such accidents, though there is no explicit attack on the security, we propose a *Symmetric Cryptographic* model, where any proprietary symmetric key algorithm can fit in. The section *[2.2]* describes this approach below.

2.2 Symmetric Cryptography Approach in Small Computing Environnements

In a small computing environment using an MCU with limited program execution memory and slow clock frequency, the thesis strongly discourages use of Asymmetric cryptography, since it would have made a sluggish response time that the device would have been literally worthless in some operations requiring instant action. Though asymmetric cryptography is preferred over a symmetric approach, user of such small computing devices must compromise to use a symmetric approach given the low-end hardware of the device and the little cost they afford for the device. Sectors or individuals who need deeper levels of security must afford higher configuration systems. In our model aimed to serve small computational environments, it is assumed that the smart phone[*] and the Bluetooth device both have knowledge of the following:

a) ***F(X,Y)***: a mathematical function which has a unique value for each pair of values to X and Y. So, we have

$$\mathbf{F(x_1, y_1) \neq F(x_2, y_2);} \text{ where either } \mathbf{x_1 \neq x_2 \text{ or } y_1 \neq y_2 \text{ or both}}$$

b) **g(X):** a mathematical function which produces a unique value for each value of X. So, we have,

$$\mathbf{g(x_1)}\,) \neq \mathbf{g(x_2)} \text{ for } x_1 \neq x_2$$

Since Bluetooth Serial Port Profile is initiated with the pairing *PIN* only and the MCU will not have enough memory to keep a list of paired devices, the device will offer an open access ***BTSPP*** to the *slave* devices. The smart phone hence must be authorized with the device in a 3-way Challenge-Response-ACK model. The smart phone first connects to the open access ***BTSPP*** and makes the following computations:

a) K_1 := F(Bluetooth_Addr, IMSI);
b) K_2: = g(*PIN*) where PIN is a changeable, setup-time n-digit (Eg. 4 digit) number provided by the user of the smart phone.
c) $K_3 := K_1 \oplus K_2$
d) Sends the following UTF-8 steam to the device:
e)

#bt=<Bluetooth Address>&imsi=<IMSI>&cmd=<COMMAND>&key =K$_3$

[Table.3] shows a list of UTF steam parser notations. However, an implementation of our engaged model is free to use any parser notations.

The device, on receiving the UTF stream, extracts the following fields

a. Bluetooth MAC address of the smart phone
b. *IMSI* of the mobile network being used in the smart phone
c. The command requested the execution of by the smart phone
d. the key, whose value is $\boldsymbol{K_3}$

The device computes **K_1 := F(Bluetooth_Addr, IMSI)** and sends a challenge UTF-8 stream as **?PIN=**

The smart phone responds to the challenge with the UTF-8 stream: **PIN=<PIN>** ; without the "<>". Let us say, the value of PIN is ***Q***

The device receives the challenge and carries out the following computations in order to authorize or in-authorize the smart phone's request to execute the command:

a) $K := K_1 \oplus K_3$ (1)
b) $R := g(\boldsymbol{Q})$ (2)
c) **IF *(K=R)* Then**

Execute the command as requested for by the smart phone

Else

Drop the request treating the smart phone to be unauthorized and disconnect from the ***BTSPP***

End IF

The entire process is described in **[Table.2]** and the timing diagram is shown in **[Fig. 1]**.

However, the thesis must encourage the use of Asymmetric cryptography for devices which are paramount to attackers for either Defense, Political or commercial rivalry or some other delicate issues and require deeper levels of security. Owners of such devices must, therefore, afford high end devices which have sufficient execution memory and clock frequency to carry out the bulk computations to achieve stronger the security and satisfactory responsiveness together. The section [2.3] below describes an Asymmetric Cryptography approach.

2.3 Asymmetric Approach in High-end Computing Environements

Computations related to the *Asymmetric* approach are clearly shown in **[Table.4]**; and the timing diagram is shown in **[Fig.2]**. The advantages that the thesis proposes to provide with to this asymmetric approach is discussed in section [3].

III. Vulnerability Analysis of Asymmetric Approach

There are broadly four possible ways that an open access Bluetooth Serial Port Profile may fall prey to an attack:

i. Another Bluetooth gadget, such as another smartphone, might try to make an attack in order to execute some commands at the device. Intention for such an attack may result from some sort of organizational, political conflict.
ii. An *evil* Bluetooth stack may spoof as the device to befool an authorized smartphone, resulting in denial of service (*DoS*), so that the original device does not respond to an emergency command coming from the smartphone.
iii. An *evil* smartphone might keep the device busy so that the genuine smartphone does not get hold of it while in a sheer urgency to execute a command. This is also a Denial of Service attack.
iv. An *evil* smartphone might intercept between an ongoing 3-way traffic in order to intrude in to the security system to make a successful attack at a later time.

3.1 Prevention of Denial of Service (DoS) Attack

Bluetooth **(IEEE 802.15)** serial port profile has already an intrinsic anti-spoofing security mechanism that suffices for normal to average attackers. To add to this inbuilt security mechanism, the proposed model offers to pass a communication parameter or a sequence of parameters delimited by a UTF-8 character, to the UTF stream, and then requires the smartphone to encrypt it with its private key. A sample of this UTF stream is already illustrated in Algorithm 2 in **[Table.4]**, where an encryption now is carried out as:

$$\mathbf{U_{CHALLENGE} := E_{+Device}(\, U_{CHALLENGE})} \qquad (3)$$

If the intercepting smartphone replays denial of service attacks by junk responses, the device will carry out the following:

a) Decrypt the UTF stream with own private key
b) Extract the pieces of encrypted information delimited by the predefined delimiter (‘**&**’) as shown in **[Table.4]**
c) Decrypt the information with the public key of the smart phone
d) If at least any one of the information thus decrypted does not match the desired value, it is not treated as a response and hence the true response stream can reach the device.

We observe an evil smartphone must fulfil all the constraints below to succeed in playing a *DoS* attack:

a) The attacker must have the private key of the device (**it’s quite absurd**)
b) It must know the UTF-8 stream formats together with the string delimiters.
c) It should have the public key of the smart phone (this is highly probable).

The thesis proposed to append two fields to control string (UTF-8 stream):
a) *IMSI* of the smartphone’s connection and b) *Bluetooth Address*; both of which should be encrypted with the private key of the smart phone.
This ensures the authenticity and integrity of the communication.
So, the smartphone appends the piece of UTF-8 to the UTF stream that it sends in a phase of communication with the device: **&IMSI=I&bt=B**
Where,

- I =$E_{\mathbf{-Phone}}$ (IMSI)
- B= $E_{\mathbf{-Phone}}$(Bluetooth Address of the smartphone)
- After this is appended to the UTF stream, the entire UTF-8 undergoes another phase of encryption

$$U^C = E_{\mathbf{+Device}}(U).$$

The above steps take care of confidentiality, authenticity and integrity of the communication and *DoS* attacks cannot succeed since successive misleading inbound communication are blocked by blacklisting attacking devices. *The standard IEEE 802.15 stack take care of this.* **[Fig.3]** illustrates the process of preventing *DoS* attacks.

IV. Experimental Illustration of the Model

The thesis not only proposes the security model and analyses its pros-and-cons, but also provides a robust, impelling implementation developed for an Android Smartphones, which communicates to a Bluetooth device running a firmware, abiding by the proposed model. Following the development process the data traffic between the devices over Bluetooth link were logged with respective timestamps to analyse the performance of the architecture. The author encloses video, stills and the applications binaries to establish this model, stating how it fits to certain systems requiring an additional security as illustrated in section [***2.1***].

4.1 Implementation of the Security Model

The implementation has two parts:

i. The development of the model on the RTOS
ii. The development of the model on an Android smartphone.

For implementation of the former, the author used Java with JSR-82 support. For the development of the second model, the author used native android development.

Listed in **[Table.7]** are the details about the systems information of the smartphone as well as the Bluetooth device. Devices have been categorized broadly as Type-S and Type-W. Type-S represents devices equipped with high end hardware and running an RTOS whilst Type-W represents simple MCU-driven Bluetooth units with little computational resources.

4.2 Analysis of Data Traffic in a Small Computing Environment

[Table.8] shows an analysis of experimental traffic between the smartphone and a small computing device, conforming to *ALGORITHM 1* described in **[Table.2]**. This experiment was carried out with a Type-W device specified in **[Table.7]**. We find that the average time for an authentication cycle of ***1177 milliseconds***. The curve we plotted is shown in **[Table.9]**.

V. Some Useful Real Life Solution Benefiting from This Thesis

We have plentiful of real life product s and solutions that could benefit from this. At the time of composing this thesis, the first author already implemented the security model he proposes through this thesis for various manufacturing companies he is consulted by in his own research lab and household, as well. A few of such successful solutions are mentioned and described below…

5.1 Router Re-Setter Mobile Computing Solution

For those who distribute a broadband Internet connection with a Wi–Fi router, a very common nuisance is having to turn off the Wi–Fi router and turn it on again after a few seconds each time their Internet seems to have become non-responsive. This process in technically known as "soft-resetting". Proposed remedy to this solution was not to dig into the root cause of this problem and rectify the issue; because there are some hundreds of manufactures of these routers and *ADSL* modems. Rather, the scope of the product was to help people soft-reset the router without having to physically move. For an aged person, who may be browsing the Internet from her bed room in the ground floor, it'd be very uneasy and painful to scale the stairs to the first floor or the terrace where the modem and router might have been placed. As a solution to this he implemented an application for smartphones to communicate to a low-cost MCU with a Bluetooth transceiver and an electrical relay to power off/on the Wi-Fi router .Schematic of such a circuitry used by the author and screen shot of the android application that controls this unit are shown in **[Fig. 4]**. Now consider that her neighbour might have installed a similar solution from the same manufacturer. From the manufacturer's view, it will not be quite feasible to distribute different smartphone applications to control different routers. A kid may thus cause disruption of Internet service by turning old lady's router off using an application installed on his mother's smartphone. To prevent this with minimum infrastructure, the security approach described in *Algorithm 1* **[Table.2]**, has been adopted. This was a real life solution providing the user with basic, *civilian* security. This solution may, however, need a better security mechanism, if the user were no ordinary person but a socially, politically or professionally important person; whose activity might have conflict of interest with certain groups of people or rivals, who might have, therefore, wanted to interrupt his work.

Use of better security mechanism, described in **[Table.4]**, however, needs better computing resources, as discussed in earlier sections of this thesis.

5.2 Smart Micro Oven Solution

Aimed at aged, ailing and carrying women, smart micro oven solution is a smartphone assisted micro-oven controller , which enables, though a competent dashboard on the smartphone GUI, easy controlling of the micro oven, like controlling the temperature, cooking interval, gaining basic information about the food being cooked, *etc*. This solution needs basic security.

5.3 Smart Phone Assisted Car Locking System

The author implemented the system in his own car, through which it is possible to lock or unlock a car, and slide up a window, and apart from this, the solution offers to provide the user with information like

- ✓ Fuel information and low-fuel indication
- ✓ Temperature of the engine
- ✓ Pollution level of the engine
- ✓ Control the heater and air conditioner

This system must be highly secured, and it accords to *Algorithm 4*

VI. Algorithms and Protocol Architecture

The device and the smart phone application have in common knowledge of the following:

1. A mathematical function F(X,Y)
2. A mathematical function g(X)
3. A predefined command or instruction set; like REQUEST_STATUS, POWER_ON, and POWER_OFF etc.

6.1 Generalized Device Authentication Algorithm

Since, this Abstraction Algorithm aims to offer a generalized solution, it does not assume any device configuration with which the smart phone will communicate. There may be a variety of devices with different configurations; like a device with an MCU and a Bluetooth (IEEE 802.15) chip; which does not have much resources to carry out complex computations that the Authentication Algorithm might involve:

a) A device with an MCU and a Bluetooth (IEEE 802.15) chip; which does not have much resources to carry out complex computations that the Authentication Algorithm might involve

b) The device may be running a firmware which can carry out complex computations but its processing speed may be too slow in responsiveness

c) A high end device might run an OS with enough resources to be commensurate with complex algorithms

Clearly, it is implied that the device being used in a system must be manufactured as per the degree of security and responsiveness needed. The thesis, therefore, aims to fit well with all these variety of devices and requirements. The thesis proposes different authentication mechanisms having different complexities, described in the tables; **[Table.2]** to **[Table.3]**

6.2 Algorithm-1: Device Authentication Model in a Small Computing Environment

The authentication mechanism is a 3-way augmentation of the Bluetooth stack.

Request-to-Connect by smartphone Bluetooth stack:

The smart phone requests execution of a command after connecting to the device over *btspp*

a) The smart phone computes K_1 as $K_1 := F(Bluetooth\_Addr, IMSI)$; where Bluetooth_Addr= the 48-bit MAC address of the Bluetooth adapter of the smart phone that requests connection ; and IMSI is belongs to the International Mobile Subscriber Identity of the connection in use by the smart phone

b) $K_2 := \mathbf{g}(PIN)$ where *PIN* is a changeable, setup-time n-digit (Eg. 4 digit) number provided by the user of the smart phone.

c) $K_3 := K_1 \oplus K_2$

The smart phone sends a UTF stream protocoled as:

#bt=<Bluetooth Address>&imsi=<IMSI>&cmd=<COMMAND>&key=K $_3$

The authentication at the device

1. The device receives the UTF and extracts the following fields

   e. Bluetooth MAC address of the smart phone
   f. IMSI of the mobile network being used in the smart phone
   g. The command that was requested for execution
   h. the key, whose value is K_3

2. Since the device does know the definition of ***F(X,Y),*** it easily computes $K_1 := F(Bluetooth\_Addr, IMSI)$
3. The device sends an application layer protocol UTF stream **?PIN=**
4. The smart phone gets this UTF-8 stream and acknowledges as **?PIN=<*PIN*>**
5. The device receives the *PIN* and computes $R := g(PIN)$
6. The device calculates *K* as

$$K := K_3 \oplus K_1$$

7. **IF** $K = R$ **THEN**

a. Allow connection and execute the command as desired

ELSE

a. Decline execution of the execution of the command desired by the smartphone

END IF

6.3 Use of UTF-8 Stream Parsers

A. '*#*' at the beginning of the UTF stream denotes that the UTF corresponds to the request from the smart phone

B. ***'?'*** at the beginning of the UTF stream denotes exchanging parameters delimited by ***'&'*** or querying a parameter by either side.

C. Though using XML or JSON would have been much more efficient in parsing a UTF-8 stream, considering that the device may have considerably low Processing power and program execution memory, we discard the implementing idea of XML and JSON on the device. They can, however, work fine on smartphones, because smartphones have considerably bigger memory and faster and powerful processor, often a dual-core or multi-core one, to parse XML and JSON.

D. If the smartphone needs to fetch a lot of information from the device, such as a device transmitting a large set of information obtained from a sensor, over Bluetooth on a pull basis, we must go for XML or JSON; to keep the information structured. The smartphone processes them quickly and effectively whereas the device does not have to parse the XML or JSON; rather some basic string programming is fabricated to generate a simple, predefined XML or JSON.

E. But, intruding **XML/JSON** unnecessarily will add overhead to either end.

6.4 Algorithm-2: Asymmetric Security Model for Device Authentication in a High Computing Environment

1. The smart phone connects to the device over *btspp* and carries out the following computations

a. $\mathbf{K_1 = F(}$*Bluetooth Address, IMSI*$\mathbf{)}$

b. Generates a pseudo random number, ***PRN***

c. $\mathbf{K_2 = g(}PIN\mathbf{)} \oplus PRN$

d. $\mathbf{K_3 = K_1 \oplus K_2}$

e. Encrypt the ***IMSI, Bluetooth address,K***$_3$ and the ***PRN*** itself with the public key of the device as

i. **I:=** $E^{+Device}$ ***(IMSI)***
ii. **B:=** $E^{+Device}$ ***(Bluetooth Address)***
iii. K_4: = $E^{+Device}$ (K_3)
iv. **J:=** $E^{+Device}$ (***PRN***)

f. **Sends the UTF stream(U) as:**

i. **?cmd=<*Cmd*> & bt=*B* & imsi=*I* & k=K_4 & R=*J***

Here,

***Cmd*= (*command_to_execute*)**

g. The **UTF** stream (**U**) is encrypted using the private key of the Phone in order to ensure authenticity of the connection: $\mathbf{U_1 = E^{-Phone}(U)}$ which is sent to the device

2. The device receives the **UTF** stream and the following steps are carried out inside the firmware of the device to authenticate and authorize the execution of a command that the smart phone had requested for;

a. The UTF stream (U) is decrypted using the public key of the Phone, which is available to the firmware of the device.

$$\mathbf{U = E^{+Phone}\ (U_1)}$$

b. Parses the UTF stream, to extract the following fields:

FIE	VALUE	IMPLICATION OF FIELD
Cmd	Command requested	The command that the Phone requested to get executed
Bt	**B**	Encrypted Bluetooth Hardware address of the smart phone's Bluetooth adapter
Imsi	**I**	Encrypted IMSI of the connection used by the smart phone
K	$\mathbf{K_4}$	Encrypted value of K_3
R	**J**	Encrypted pseudo random number

c. The device then decrypts these fields with its *private key*; shown below:

FIELD	DECRYPTION PROCESS
cmd	This field was not encrypted by the smart phone; so decryption is not needed
bt	Bluetooth Address of Smart Phone = $\mathbf{E^{-device}(B)}$
imsi	$\mathbf{IMSI = E^{-device}(I)}$
k	$\mathbf{K_3 = E^{-device}(K_4)}$
R	$\mathbf{PRN = = E^{-device}(J)}$

d. These decrypted values of the Bluetooth Address, IMSI and the Pseudo random number are used to carry out the following computations to authenticate (and, authorize) the device

i. $\mathbf{K_1 = F(\mathit{Bluetooth\_Address, IMSI})}$
ii. $\mathbf{K_2 = K_1 \oplus K_3}$

e. The device ***does*** know that, $\mathbf{K_2 = g(\mathit{PIN}) \oplus PRN}$; where ***PRN*** is known to it; so it can authenticate the smart phone simply based on getting the right value of ***PIN***, which it can ask the smart phone to send as response; because, the device also knows the definition of the function ***g***(X) which produces a unique value for each value of X=x.

f. Generates a pseudo random number, ***Q*** , which is used as a challenge

g. The challenge Q is encrypted using the public key of the smart phone;

$$\mathbf{C: = E^{+Phone}\ (Q)}$$

h. The device now generates as a challenge the UTF stream ($\mathbf{U_{CHALLENGE}}$) expressed as ***?CHALLENGE=<C>,*** which is further encrypted with the private key of the device; given as

$$\mathbf{U_{CHALLENGE} := E^{-Device}(\ U_{CHALLENGE})}$$

3. The smart phone, on decrypting the UTF stream with the public key of the device, decrypts the challenge with its own private keys and responds to the challenge with the UTF, $\mathbf{U_{RESPONSE}}$, encrypted with its private key; as follows:

a. $U_{CHALLENGE} := E^{+Device}(\ U_{CHALLENGE})$
b. $\mathbf{Q:= E^{-Phone}\ (C)}$
c. Calculates the response, **RESPONSE,** as **RESPONSE:= Q ⊕ PIN**
d. Encrypts the response with its private key and responds to the challenge with the UTF stream $\mathbf{U_{RESPONSE}}$**:= *?RESPONSE=<RSP>*** where $\mathbf{RSP:= E^{-Phone}\ (RESPONSE)}$
e. $\mathbf{U_{RESPONSE} := E^{+Device}\ (U_{RESPONSE})}$

4. The device receives the response and authenticates the smart phone based on the sequence of computations that follow below:

a. $\mathbf{U_{RESPONSE}:= E^{-Device}\ (U_{RESPONSE})}$
b. Decrypts the response value with the public key of the smart phone $\mathbf{RESPONSE:= E^{+Phone}\ (RSP)}$
c. Since the device has both the values of Q and REPONSE, it can readily find out the value of the *PIN* as

$$\mathbf{PIN= Q \oplus RESPONSE}$$

[∵ *XOR is reversible*]

d. $\mathbf{A= g(\mathit{PIN}) \oplus PRN}$
e. It can compare, in a sequel to the computations in **Step-2.d** the value of K_2 with the value of ***A*** to verify if the value equals the value of K_2, since $\mathbf{K_2:= g(\mathit{PIN}) \oplus PRN}$

f. **IF (A=K_2) Then**
Executes the command which was requested by the smart phone
Else

- Simply drop request and terminate the connection
- Log the rejection information and send it to the device administrator over any media such as SMS, Instant Message over **XMPP**, Mail, or other sort of physical alarms like a siren

End IF

VII. Tables and Figures

7.1 Table.1: Generalized Device Authentication Algorithm

Since, this **Table** aims to offer a generalized solution, it does not assume any device configuration with which the smart phone will communicate. There may be a variety of devices with different configurations; like

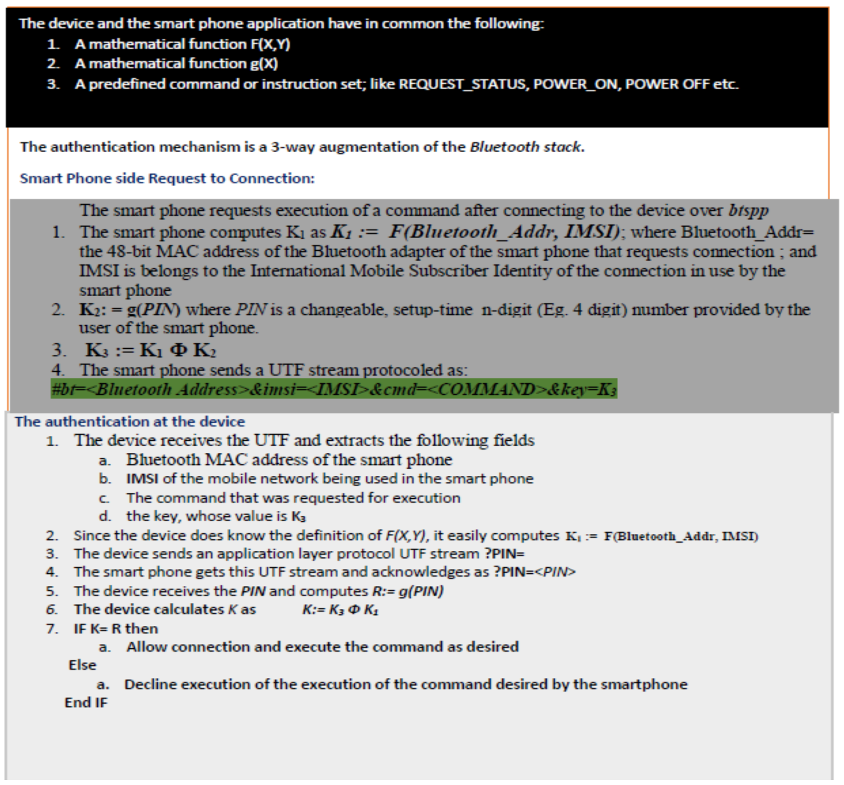
a) A device with an *MCU* and a Bluetooth (*IEEE 802.15*) chip; which does not have much resources to carry out complex computations that the Authentication Algorithm might involve
b) The device may be running a firmware which can carry out complex computations but its processing speed may be too slow in responsiveness
c) A high end device might run an OS with enough resources to be commensurate with complex algorithms

Clearly, it is implied that the device being used in a system must be manufactured as per the degree of security and responsiveness needed. The thesis, therefore, aims to fit well with all these variety of devices and requirements. The thesis proposes different authentication mechanisms having different complexities, described in the tables; **[Table.2]** to **[Table.3].**

1. The device and smart phone are connected over Bluetooth Serial Port Profile
2. The smart phone application sends a REQUEST_CONTROL packet (UTF stream)
3. The Device receives the Request and processes it using a suitable ALGORTIHM and checks if the smart phone application is authorized to communicate
    a. If it is, it is allowed to continue communication;
    b. Else it is denied communication and optionally added to the roster of 'Black Smart phones'

7.2 TABLE.2

ALGORITHM 1: DEVICE AUTHENTICATION ALGORITHM MODEL IN A SMALL COMPUTING ENVIRONMENT



The device and the smart phone application have in common the following:
1. A mathematical function F(X,Y)
2. A mathematical function g(X)
3. A predefined command or instruction set; like REQUEST_STATUS, POWER_ON, POWER OFF etc.

The authentication mechanism is a 3-way augmentation of the *Bluetooth stack.*

Smart Phone side Request to Connection:

```
   The smart phone requests execution of a command after connecting to the device over btspp
1. The smart phone computes K1 as K1 := F(Bluetooth_Addr, IMSI); where Bluetooth_Addr=
   the 48-bit MAC address of the Bluetooth adapter of the smart phone that requests connection ; and
   IMSI is belongs to the International Mobile Subscriber Identity of the connection in use by the
   smart phone
2. K2: = g(PIN) where PIN is a changeable, setup-time n-digit (Eg. 4 digit) number provided by the
   user of the smart phone.
3. K3 := K1 Φ K2
4. The smart phone sends a UTF stream protocoled as:
#bt=<Bluetooth Address>&imsi=<IMSI>&cmd=<COMMAND>&key=K3
```

The authentication at the device

```
1. The device receives the UTF and extracts the following fields
      a. Bluetooth MAC address of the smart phone
      b. IMSI of the mobile network being used in the smart phone
      c. The command that was requested for execution
      d. the key, whose value is K3
2. Since the device does know the definition of F(X,Y), it easily computes K1 := F(Bluetooth_Addr, IMSI)
3. The device sends an application layer protocol UTF stream ?PIN=
4. The smart phone gets this UTF stream and acknowledges as ?PIN=<PIN>
5. The device receives the PIN and computes R:= g(PIN)
6. The device calculates K as      K:= K3 Φ K1
7. IF K= R then
      a. Allow connection and execute the command as desired
   Else
      a. Decline execution of the execution of the command desired by the smartphone
   End IF
```

7.3 Table.3: Utf Stream Parsing Notations

A. ‘*#*’ at the beginning of the UTF stream denotes that the UTF corresponds to the request from the smart phone
B. ***‘?’*** at the beginning of the UTF stream denotes exchanging parameters delimited by ‘***&’*** or querying a parameter by either side.
C. Though using XML or JSON would have been much more efficient in parsing a UTF-8 stream, considering that the device may have considerably low Processing power and program execution memory, we discard the implementing idea of XML and JSON on the device. They can, however, work fine on smartphones, because smartphones have considerably bigger memory and faster and powerful processor, often a dual-core or multi-core one, to parse XML and JSON.
D. If the smartphone needs to fetch a lot of information from the device, such as a device transmitting a large set of information obtained from a sensor, over Bluetooth on a pull basis, we must go for XML or JSON; to keep the information structured. The smartphone processes them quickly and effectively whereas the device does not have to parse the XML or JSON; rather some basic string programming is fabricated to generate a simple, predefined XML or JSON.
E. But, intruding XML/JSON unnecessarily will add overhead to either end.

7.4 Table.4:

Algorithm 2: Asymmetric Security Model For Device Authentication In A High Computing Environment

Algorithm 2: Asymmetric Security Model for Device Authentication	
Entities Available to The Device:	
1	**Public key of the Smart phone application (via PKI)**
2	**Own Public and Private keys**
Entities available to the Smart Phone:	
1	**Public key of the Device (via *PKI*)**
2	**Own Public and Private keys**
3	**Hardware Address of its own Bluetooth adapter**
4	***IMSI* of itself (The GSM connection it bears)**
5	**A one-time setup, changeable PIN**
Entities available to both the device and the Smart phone application:	
1	**F(X,Y)** : a mathematical function that produces unique value for each pair of {x,y}
2	**g(X)**: a mathematical function which produces unique value for a given X=x
Notations used in computational illustrations:	
1	$\mathbf{E}^{+}$: Encryption (or decryption) with public key
2	$\mathbf{E}^{-}$: Encryption (or decryption) with private key
3	$\mathbf{E}^{+W}$: Encryption (or decryption) with public key of **W** where ***W Є {Device, Phone}***
4	$\mathbf{E}^{-W}$: Encryption (or decryption) with private key of **W** where ***W Є {Device,Phone}***

1. The smart phone connects to the device over *btspp* and carries out the following computations

a. $\mathbf{K_1 = F(Bluetooth\_Address, IMSI)}$
b. Generates a pseudo random number, ***PRN***

c. $\mathbf{K_2 = g(}\boldsymbol{PIN}\mathbf{)\ \Phi\ }\boldsymbol{PRN}$
d. $\mathbf{K_3 = K_1\ \Phi\ K_2}$
e. Encrypt the ***IMSI, Bluetooth address,K_3*** and the ***PRN*** itself with the public key of the device as

i. **I:=** $E^{+\text{Device}}$ (***IMSI***)
ii. **B:=** $E^{+\text{Device}}$ (***Bluetooth Address***)
iii. ***K_4*: =** $E^{+\text{Device}}$ (***K_3***)
iv. **J:=** $E^{+\text{Device}}$ (***PRN***)

f. **Sends the UTF stream(U) as:**

ii. **?cmd=<Cmd> & bt=*B* & imsi=*I* & k=*K_4* & R=*J*** where **cmd=** $E^{+\text{Device}}$ (***command_to_execute***)

g. The UTF stream (U) is encrypted using the private key of the Phone in order to ensure authenticity of the connection: $\mathbf{U_1 = E^{-Phone}(U)}$ which is sent to the device

2. The device receives the UTF stream and the following steps are carried out inside the firmware of the device to authenticate and authorize the execution of a command that the smart phone had requested for;

a. The UTF stream (U) is decrypted using the public key of the Phone, which is available to the firmware of the device.

$$\mathbf{U = E^{+Phone}(U_1)}$$

b. Parses the UTF stream, to extract the following fields:

FIELD	VALUE	IMPLICATION OF FIELD
cmd	Command requested execution of	The command that the Phone requested to get executed
bt	**B**	Encrypted Bluetooth Hardware address of the smart phone's Bluetooth adapter
imsi	**I**	Encrypted IMSI of the connection used by the smart phone
k	$\mathbf{K_4}$	Encrypted value of ***K_3***
R	**J**	Encrypted pseudo random number

c. The device then decrypts these fields with its *private key*; shown below:

FIELD	DECRYPTION PROCESS
cmd	This field was not encrypted by the smart phone; so decryption is not needed
bt	Bluetooth Address of Smart Phone = $\mathbf{E^{-device}(B)}$
imsi	**IMSI =** $\mathbf{E^{-device}(I)}$
k	$\mathbf{K_3 = E^{-device}(K_4)}$
R	**PRN = =** $\mathbf{E^{-device}(J)}$

d. These decrypted values of the Bluetooth Address, IMSI and the Pseudo random number are used to carry out the following computations to authenticate (and, authorize) the device

i. $\mathbf{K_1 = F(}$***Bluetooth_Address, IMSI***$\mathbf{)}$
ii. $\mathbf{K_2 = K_1\ \Phi\ K_3}$

e. The device ***does*** know that, $\mathbf{K_2 = g(}\boldsymbol{PIN}\mathbf{)\ \Phi\ PRN;}$ where ***PRN*** is known to it; so it can authenticate the smart phone simply based on getting the right value of ***PIN***, which it can ask the smart phone to send as response; because, the device also knows the definition of the function ***g***(X) which produces a unique value for each value of X=x.
f. Generates a pseudo random number, ***Q*** , which is used as a challenge
g. The challenge Q is encrypted using the public key of the smart phone;

$$\mathbf{C: = E^{+Phone}\ (Q)}$$

h. The device now generates as a challenge the UTF stream (**$U_{CHALLENGE}$**) expressed as ***?CHALLENGE=<C>,*** which is further encrypted with the private key of the device; given as

$$\mathbf{U_{CHALLENGE} := E^{-Device}(\ U_{CHALLENGE})}$$

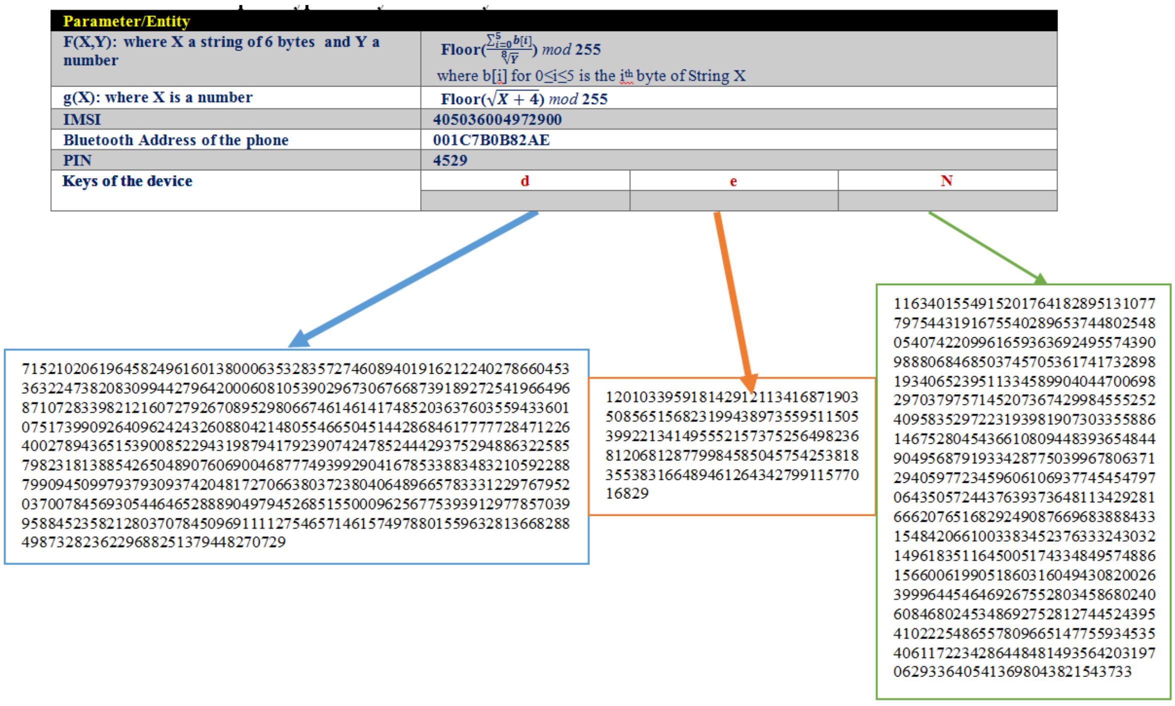
3. The smart phone, on decrypting the UTF stream with the public key of the device, decrypts the challenge with its own private keys and responds to the challenge with the UTF, **$U_{RESPONSE}$**, encrypted with its private key; as follows:
4. The device receives the response and authenticates the smart phone based on the sequence of computations that follow below:

g. **$U_{RESPONSE}$:= E**$^{-Device}$ **($U_{RESPONSE}$)**
h. Decrypts the response value with the public key of the smart phone **RESPONSE:= E^{+Phone} (RSP)**
i. Since the device has both the values of Q and REPONSE, it can readily find out the value of the ***PIN*** as **PIN= Q Φ RESPONSE** [***Since, XOR is reversible***]
j. **A= g(*PIN*) Φ PRN**
k. It can compare, in a sequel to the computations in **Step-2.d** the value of K_2 with the value of ***A*** to verify if the value equals the value of K_2, since **K_2:= g(*PIN*) Φ PRN**
l. **IF (A=K_2) Then**

```
        Executes the command which was requested by the smart phone
    Else
        ➢ Simply drop request and terminate the connection
        ➢ Log the rejection information and send it to the device administrator over any media such as
          SMS, Instant Message over XMPP, Mail, or other sort of physical alarms like a siren
    End IF
```

7.5 Table.5

Illustration 1: Simple Crypto analysis of 3-way authorization mechanism



Parameter/Entity			
F(X,Y): where X a string of 6 bytes and Y a number	$\mathrm{Floor}\left(\frac{\sum_{i=0}^{5} b[i]}{\sqrt[8]{Y}}\right) \bmod 255$ where b[i] for $0 \le i \le 5$ is the ith byte of String X		
g(X): where X is a number	$\mathrm{Floor}(\sqrt{X+4}) \bmod 255$		
IMSI	405036004972900		
Bluetooth Address of the phone	001C7B0B82AE		
PIN	4529		
Keys of the device	d	e	N

d:

7152102061964582496160138000635328357274608940191621224027866045336322473820830994427964200060810539029673067668739189272541966496871072833982121607279267089529806674614614174852036376035594336010751739909264096242432608804214805546650451442868461777772847122640027894365153900852294319879417923907424785244429375294886322585798231813885426504890760690046877749399290416785338834832105922887990945099793793093742048172706638037238040648966578333122976795203700784569305446465288890497945268515500096256775393912977857039958845235821280370784509691111275465714615749788015596328136682884987328236229688251379448270729

e:

12010339591814291211341687190350856515682319943897355951150539922134149555215737525649823681206812877998458504575425381835538316648946126434279911577016829

N:

11634015549152017641828951310777975443191675540289653744802548054074220996165936369249557439098880684685037457053617417328981934065239511334589904044700698297037975714520736742998455525240958352972231939819073033558861467528045436610809448393654844904956879193342877503996780637129405977234596061069377454547970643505724437639373648113429281666207651682924908766968388843315484206610033834523763332430321496183511645005174334849574886156600619905186031604943082002639996445464692675528034586802406084680245348692752812744524395410222548655780966514775593453540611722342864484814935642031970629336405413698043821543733

7.6 Table.6: Results obtained by Experiments:

Parameter	Proceudre	Calculated Value
K_1	=F(bt,IMSI)	211
K_2	= g(*PIN*) Φ *PRN*	2309
K_3	**= K_1 Φ K_2**	2518
B	= $E^{+Device}$ (Bluetooth Address)	D□LhøpitíÅœN µŇK;Ñ®1□™IT‘F~□ŒŒ□ø¡Ä*ÝÄœ¾ãë@÷«][□[wîÁ-ô YA2Øò6 8üÄ÷éG<¿‡?/ãNcúÇˆj‚.«□;)TA□□Â¼urù6t;ì@Ç+ß%}:mÛg)mb□□p¤…ê$□2Íe"0å°¾4ÖŸ%ö ¥ÑãÄ8Ç!³W_‘Ô@ƒÏ+□2ž.ÔF!□ŒÎ$‚?□ßz½-?û?Æ,Ú?5¬:ÝÜ¢ý□;CkÓgbð□6ncÿ□&□¼Ì%□™×¶7□ø/‹ ŽeŽ;□□`],¤ms□□8□cæ(wc□™0oêi¹ô«•á□wQN`¾,
I	= $E^{+Device}$ (IMSI)	f?µ§‚£′?;7Ù}z Qt,û‰Ž°m¼\ÖEÜæ@•Äã,ßðÖ□□óà□.NÜ□8¼ŠG ¾ó"¹2iW ŸE0ï¡-j□?_»=Ž>…›X-k"ÿ•Ï«H‡¨7□Ê;æ,-v3Ù•1¤Ùœ□ÒmBfÍ8ãB=□e□□sæ}æ□^oä$7[´µ□□eŒYA•W=□ ¶)□„ ÿ-'Ÿ.Í□øòèž<øc–ÝìàËø^0÷,æn?□%□'Ì×ÄÜ€]□‡Õ„%â÷\ÿ{"=□□ÅW¥?s□DØk¬Vp]ñQö¾]¨ m□?ÝHG□ □Ü□ZÃ„‚´÷¥"}ž□Ž□z,n□[LÙ9?-□¦È³
K_4	= $E^{+Device}$ (K_3)	r#«"°┬÷┝┘7±O)ˊÏ┘n1WÎ[»Ó• doó?cèrí›W"+□s¹ÿ©?a¸m£↕◄ß+├4 └svŽ„H'Åš†-v±}Õ©ÁýE♂hi fÇÐ¾♀├ù♀Œ┴┬$%‹0↑+ç²r●H¿:Gc¤j"EÔõiô_ÔË=ðÕ??Ëqc┐en?•«- õ┤KAÉK◄þÌy↕æâyDÇ?ªÜD♀♀¬ Ôje¹ Ä‼s7nm┤ñwù:è↑ÛÃØw~Ç→ß'sY=‡ˆï–¨ê?<5Q●6½úª°☼îƒN?DÕ$ £Ìƒ¶.ÔÐY6U?)IéJ±xa–í BÐ┐Ï'}%-U ¦┐•4?¥!ˆ¹¢\
J	= $E^{+Device}$ (PRN)	Q□ZFă□?Šˆ$$@Ä?ô¸Ê1è5¡ÀËÖÜcíF•|ECÙiÝà<0ákZ;'™¯KÀ~ö□□û,•gût7I:(öBi×-7…□*›RGÉFø2Ò□ʹ ??|¿?<Ä›'W'‹=Þþ‡–»□×kÒn Á□ùµÏ%?,5□¢È ¼?°žu¿?C*k!ó?ÚWÃ□_‘Áóú4»ÆÏÝ17Ï¦Ÿ,óF 4„ +"?K,C2ß:‰B87Í□A›Á-ä□Q?ü`æƒh$□öÝ□)Ñdö-Ì©□□Dïő,□%"|?ò6€|«Ã5□ð□ Û"S/°£,□□mÛ×7ù□›" ‰?"úþø˄á'»ŸÅpg
CMD	SHUT_DOWN	□«ÈéµRBâ©ÿOa□_□Ä‹'□□C³=f!å□□Yd ¹š[†r$ rÝš□Ü□¿□+ă9^J7¥(a\ă3□q□ð?‡ì÷□ú□›Ü)üåþ)□8r □pä□Cø =°z□æ"ãÖ®4'¯'Ò 3PÀbØnr3±z,Êˆ ÚÎž+FÔò·…□/□·™&BV¨UÞ?C÷ž5*fr□úJHdWÊ,",ăŠí"′=ì¾-ûÒ□i …CcŠòðð:ª®r-¹iê□u2)Á□Ñh1□×œË1¹ÝÝ§°4i›‡&□U߃ÜvfðÉuý5.%¾án□ñûO/¬1□r□a…gÈ□±Gë"Y$êãF ð′□õhÅ'ª□

7.7 Table.7

Systems information of the Smart phone

System Parameter	System Parameter Details
Operating System *	Android version 4.2.2
Processor*	1GHz , Single-core
RAM*	512 MB
Internal Storage	2 GB (inclusive of bundled software)
Bluetooth version*	IEEE 802.15 version 2.1
Connectivity	IEEE 802.11 b/g/n
Mobile Networks	GSM, 3G/EDGE
Sensors	Gravity sensor, Proximity sensor, accelerometer
GPS	GPS, A-GPS
Display	7.83 cm capacitive touch with 5 touch pointing

Fields that pertains to the thesis, are marked with (*)

Systems information about the Device type- S (High end computing)

System Parameter	System Parameter Details
Operating System *	Custom Linux Kernel
Processor*	800 MHz
RAM*	512 MB
Bluetooth version*	IEEE 802.15 version 2.1
Applications	Java ME support (MIDP 2.0)

Systems information about the Device type- W (Very low performance)

System Parameter	System Parameter Details	
Operating System *	No Operating System	
Processor*	AT89S51 Compatible with MCS-51	
Memory*	RAM	128 bytes
	ROM	4KB ISP Flash Memory
Bluetooth version*	***IEEE 802.15*** version 2.1 (Integrated Fire-Fly chip)	

7.8 *Table.8: Analysis of Received UTF Stream*

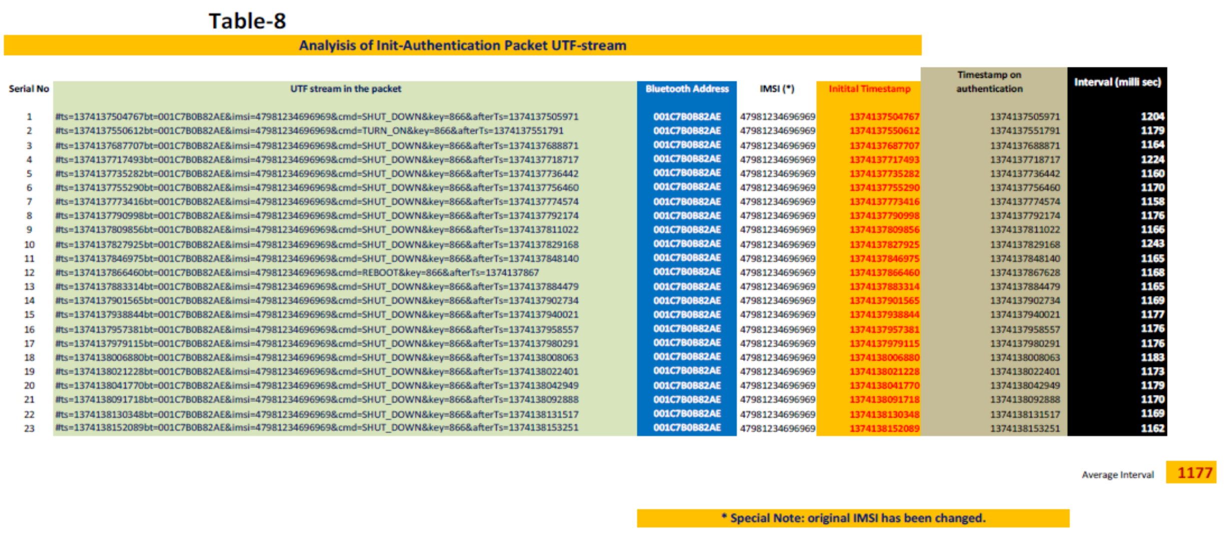


Table-8

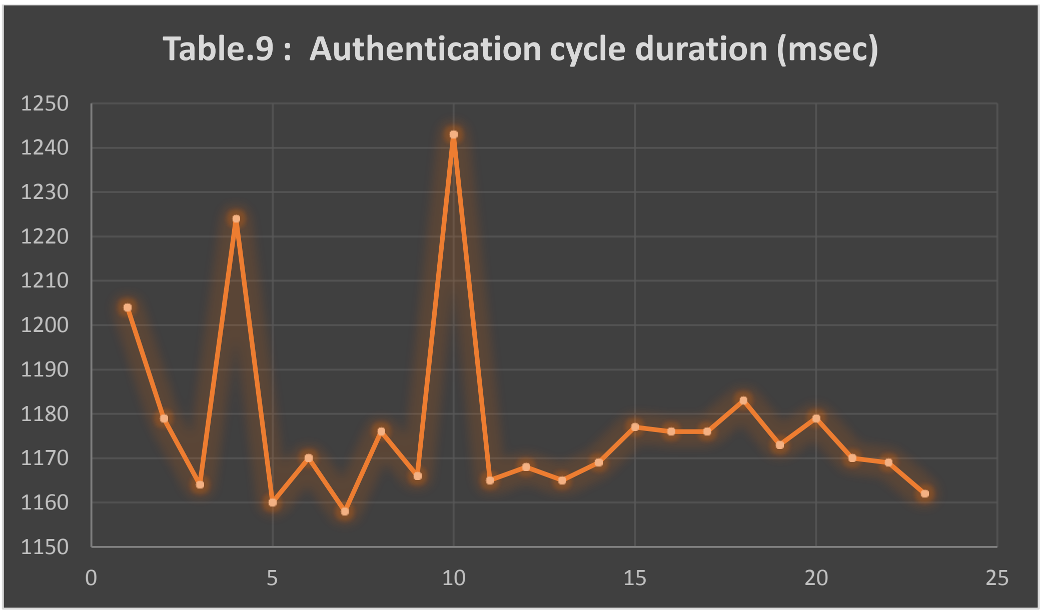
Analyisis of Init-Authentication Packet UTF-stream

Serial No	UTF stream in the packet	Bluetooth Address	IMSI (*)	Initital Timestamp	Timestamp on authentication	Interval (milli sec)
1	#ts=1374137504767bt=001C7B0B82AE&imsi=47981234696969&cmd=SHUT_DOWN&key=866&afterTs=1374137505971	001C7B0B82AE	47981234696969	1374137504767	1374137505971	1204
2	#ts=1374137550612bt=001C7B0B82AE&imsi=47981234696969&cmd=TURN_ON&key=866&afterTs=1374137551791	001C7B0B82AE	47981234696969	1374137550612	1374137551791	1179
3	#ts=1374137687707bt=001C7B0B82AE&imsi=47981234696969&cmd=SHUT_DOWN&key=866&afterTs=1374137688871	001C7B0B82AE	47981234696969	1374137687707	1374137688871	1164
4	#ts=1374137717493bt=001C7B0B82AE&imsi=47981234696969&cmd=SHUT_DOWN&key=866&afterTs=1374137718717	001C7B0B82AE	47981234696969	1374137717493	1374137718717	1224
5	#ts=1374137735282bt=001C7B0B82AE&imsi=47981234696969&cmd=SHUT_DOWN&key=866&afterTs=1374137736442	001C7B0B82AE	47981234696969	1374137735282	1374137736442	1160
6	#ts=1374137755290bt=001C7B0B82AE&imsi=47981234696969&cmd=SHUT_DOWN&key=866&afterTs=1374137756460	001C7B0B82AE	47981234696969	1374137755290	1374137756460	1170
7	#ts=1374137773416bt=001C7B0B82AE&imsi=47981234696969&cmd=SHUT_DOWN&key=866&afterTs=1374137774574	001C7B0B82AE	47981234696969	1374137773416	1374137774574	1158
8	#ts=1374137790998bt=001C7B0B82AE&imsi=47981234696969&cmd=SHUT_DOWN&key=866&afterTs=1374137792174	001C7B0B82AE	47981234696969	1374137790998	1374137792174	1176
9	#ts=1374137809856bt=001C7B0B82AE&imsi=47981234696969&cmd=SHUT_DOWN&key=866&afterTs=1374137811022	001C7B0B82AE	47981234696969	1374137809856	1374137811022	1166
10	#ts=1374137827925bt=001C7B0B82AE&imsi=47981234696969&cmd=SHUT_DOWN&key=866&afterTs=1374137829168	001C7B0B82AE	47981234696969	1374137827925	1374137829168	1243
11	#ts=1374137846975bt=001C7B0B82AE&imsi=47981234696969&cmd=SHUT_DOWN&key=866&afterTs=1374137848140	001C7B0B82AE	47981234696969	1374137846975	1374137848140	1165
12	#ts=1374137866460bt=001C7B0B82AE&imsi=47981234696969&cmd=REBOOT&key=866&afterTs=1374137867	001C7B0B82AE	47981234696969	1374137866460	1374137867628	1168
13	#ts=1374137883314bt=001C7B0B82AE&imsi=47981234696969&cmd=SHUT_DOWN&key=866&afterTs=1374137884479	001C7B0B82AE	47981234696969	1374137883314	1374137884479	1165
14	#ts=1374137901565bt=001C7B0B82AE&imsi=47981234696969&cmd=SHUT_DOWN&key=866&afterTs=1374137902734	001C7B0B82AE	47981234696969	1374137901565	1374137902734	1169
15	#ts=1374137938844bt=001C7B0B82AE&imsi=47981234696969&cmd=SHUT_DOWN&key=866&afterTs=1374137940021	001C7B0B82AE	47981234696969	1374137938844	1374137940021	1177
16	#ts=1374137957381bt=001C7B0B82AE&imsi=47981234696969&cmd=SHUT_DOWN&key=866&afterTs=1374137958557	001C7B0B82AE	47981234696969	1374137957381	1374137958557	1176
17	#ts=1374137979115bt=001C7B0B82AE&imsi=47981234696969&cmd=SHUT_DOWN&key=866&afterTs=1374137980291	001C7B0B82AE	47981234696969	1374137979115	1374137980291	1176
18	#ts=1374138006880bt=001C7B0B82AE&imsi=47981234696969&cmd=SHUT_DOWN&key=866&afterTs=1374138008063	001C7B0B82AE	47981234696969	1374138006880	1374138008063	1183
19	#ts=1374138021228bt=001C7B0B82AE&imsi=47981234696969&cmd=SHUT_DOWN&key=866&afterTs=1374138022401	001C7B0B82AE	47981234696969	1374138021228	1374138022401	1173
20	#ts=1374138041770bt=001C7B0B82AE&imsi=47981234696969&cmd=SHUT_DOWN&key=866&afterTs=1374138042949	001C7B0B82AE	47981234696969	1374138041770	1374138042949	1179
21	#ts=1374138091718bt=001C7B0B82AE&imsi=47981234696969&cmd=SHUT_DOWN&key=866&afterTs=1374138092888	001C7B0B82AE	47981234696969	1374138091718	1374138092888	1170
22	#ts=1374138130348bt=001C7B0B82AE&imsi=47981234696969&cmd=SHUT_DOWN&key=866&afterTs=1374138131517	001C7B0B82AE	47981234696969	1374138130348	1374138131517	1169
23	#ts=1374138152089bt=001C7B0B82AE&imsi=47981234696969&cmd=SHUT_DOWN&key=866&afterTs=1374138153251	001C7B0B82AE	47981234696969	1374138152089	1374138153251	1162

Average Interval 1177

* Special Note: original IMSI has been changed.

7.9 *Table.9: Analysis of Received UTF Stream*



7.10 Fig.1: Security Protocol

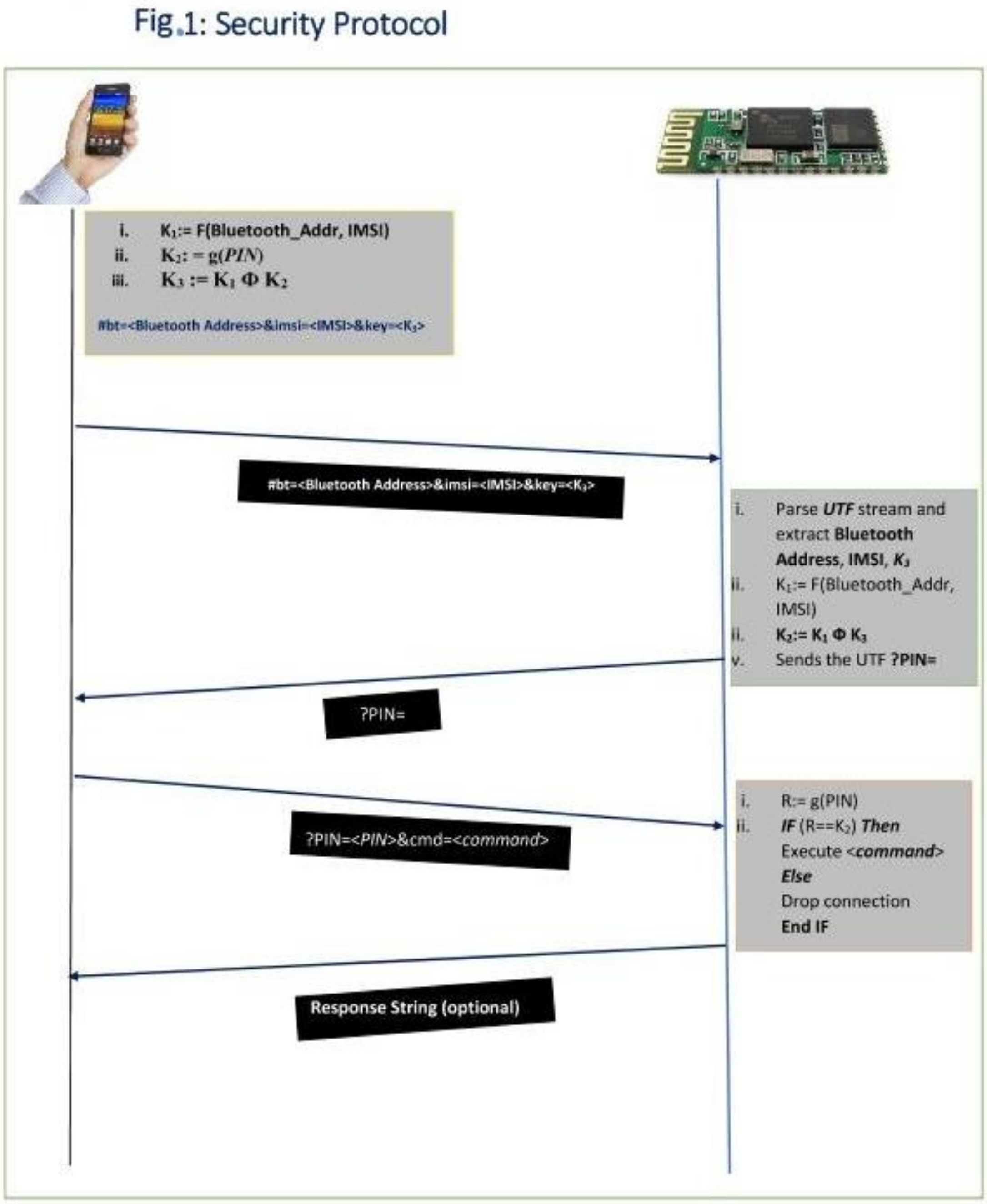
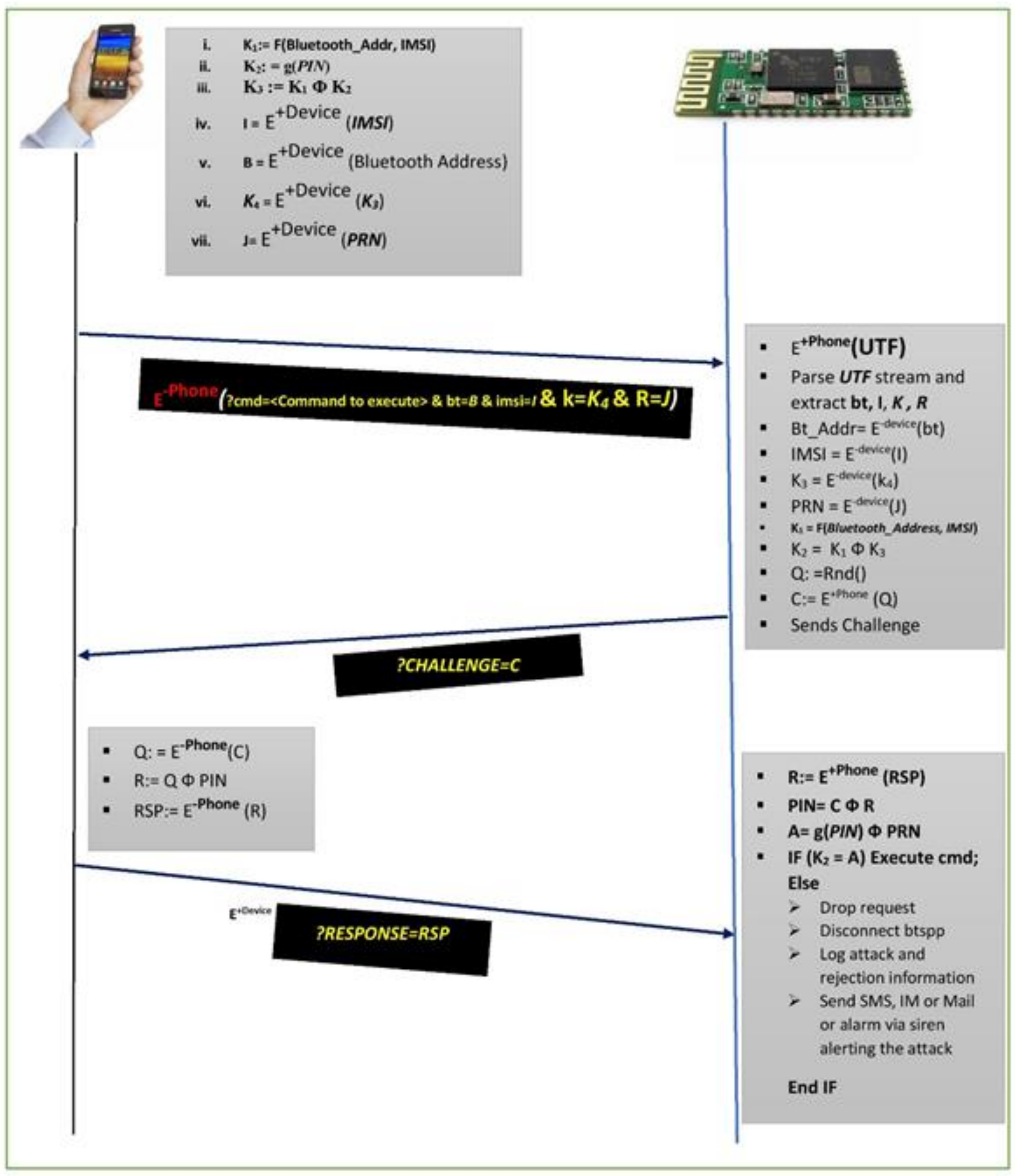


Fig.1: Security Protocol
i. K₁:= F(Bluetooth_Addr, IMSI)
ii. K₂: = g(PIN)
iii. K₃ := K₁ Φ K₂
#bt=<Bluetooth Address>&imsi=<IMSI>&key=<K₃>
#bt=<Bluetooth Address>&imsi=<IMSI>&key=<K₃>
i. Parse UTF stream and extract Bluetooth Address, IMSI, K₃
ii. K₁:= F(Bluetooth_Addr, IMSI)
ii. K₂:= K₁ Φ K₃
v. Sends the UTF ?PIN=
?PIN=
?PIN=<PIN>&cmd=<command>
i. R:= g(PIN)
ii. IF (R==K₂) Then Execute <command> Else Drop connection End IF
Response String (optional)

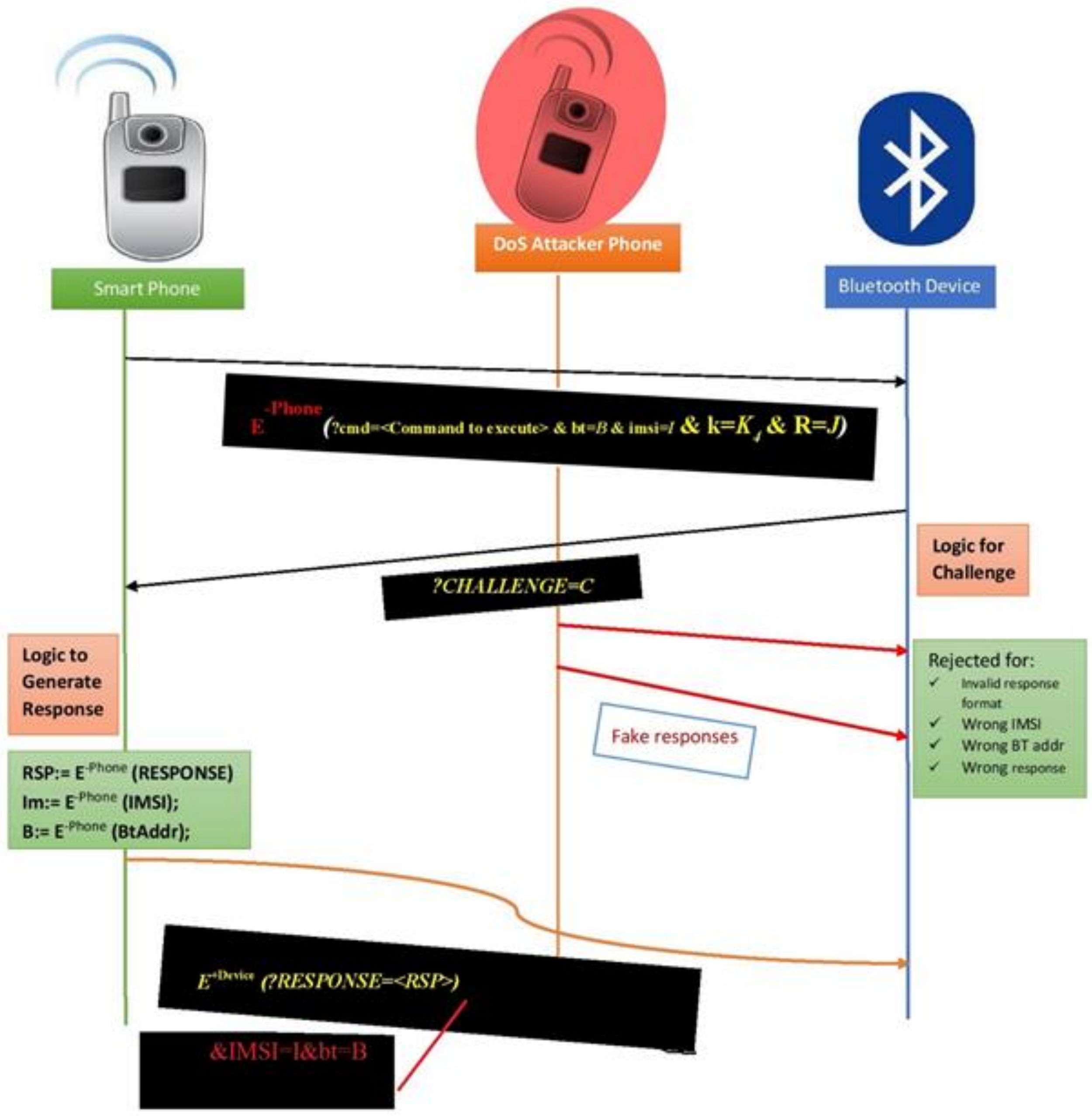
7.11 Fig.2: Asymmetric Security Model for High end Devices

Fig .2: Asymmetric Security Model (HIGH-END DEVICE)


i. K_1:= F(Bluetooth_Addr, IMSI)
ii. K_2: = g(*PIN*)
iii. K_3 := K_1 Φ K_2
iv. I = $E^{+Device}$ (*IMSI*)
v. B = $E^{+Device}$ (Bluetooth Address)
vi. K_4 = $E^{+Device}$ (K_3)
vii. J= $E^{+Device}$ (*PRN*)
E^{-Phone}(?cmd=<Command to execute> & bt=*B* & imsi=*I* & k=K_4 & R=*J*)
E^{+Phone}(UTF)
Parse *UTF* stream and extract **bt,** I, ***K*** , ***R***
Bt_Addr= $E^{-device}$(bt)
IMSI = $E^{-device}$(I)
K_3 = $E^{-device}$(k_4)
PRN = $E^{-device}$(J)
K_1 = F(*Bluetooth_Address, IMSI*)
K_2 = K_1 Φ K_3
Q: =Rnd()
C:= E^{+Phone} (Q)
Sends Challenge
?CHALLENGE=C
Q: = E^{-Phone}(C)
R:= Q Φ PIN
RSP:= E^{-Phone} (R)
$E^{+Device}$
?RESPONSE=RSP
R:= E^{+Phone} (RSP)
PIN= C Φ R
A= g(*PIN*) Φ PRN
IF (K_2 = A) Execute cmd;
Else
Drop request
Disconnect btspp
Log attack and rejection information
Send SMS, IM or Mail or alarm via siren alerting the attack
End IF

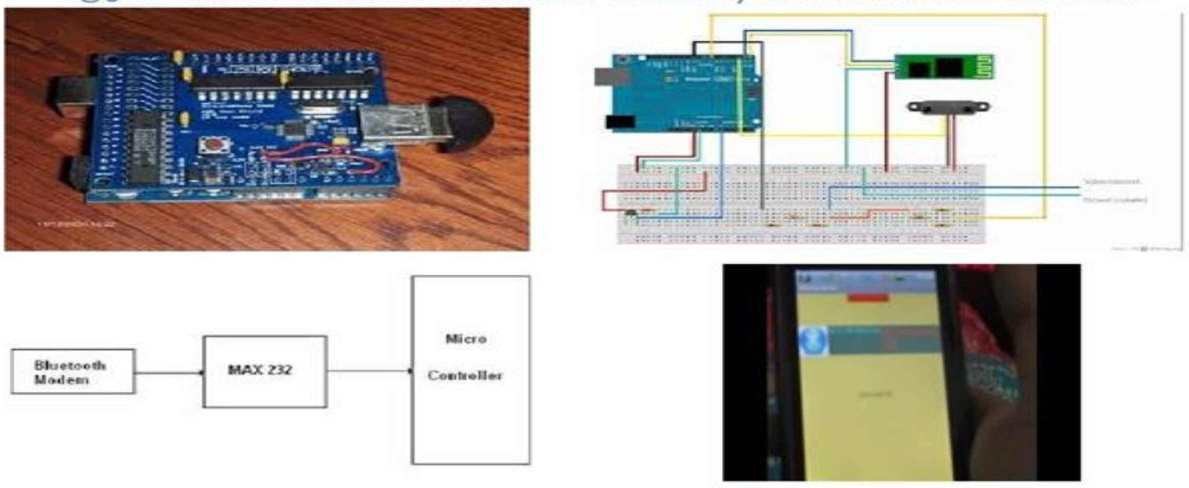
7.12 Fig.3: Prevention of DoS Attack

Fig.3: Prevention of Denial of Service Attacks



7.13 Fig.4: Circuitry and Screen-shot of Router Re-setter Solution

Fig.4: Router re-setter solution Circuitry & Mobile screen shot



VIII. CONCLUSION AND FUTURE SCOPE

The proposed model does not assume any specific cryptographic algorithm; rather aims to offer a generic abstraction over Bluetooth security model. It not only addresses the hardware diversity of devices, but also offers a Link layer implementation for achieving augmented security mechanism with a standard Bluetooth stack. Since our model does not dig into the Link layer but provides an abstraction over the application layer, it equally applies to any peer-to-peer wireless communication protocol that demands security for instrumentation. Even at its present state it can hold good for Wi-Fi Direct technology, which is widely in use in smart televisions and phones. Since Wi-Fi direct has several advantages over Bluetooth, it may replace Bluetooth very soon and our proposed model will still hold valid with whatever peer-to-peer wireless communication protocol may rule.

REFERENCES

[1] J .Padgette, K.Scarfone, L.Chen "Guide to Bluetooth Security", Computer Security N IST, 2012 http://csrc.nist.gov/publications/nistpubs/800-97/SP800-97.pdf

[2] M.L. Das, R. Mukkamala, "Revisit in g Bluetooth Security", ICISS 2008, LNCS 5352 pp. 132-139

[3] Mohammed Mana, Mohammed Feham, and Boucif Amar Bensaber, "A light weight protocol to provide location privacy in wireless body area networks", International Journal of Network Security and its Applications (IJNSA), Vol.3, No.2, March 2011

[4] "Going Around with Bluetooth in Full Safety", International Journal of Distributed and Parallel Systems (IJDPS) Vol.3, No.1, January 2012

[5] Nateq Be-Nazir Ibn Minar, Mohammed Tarique, "Bluetooth security threats and solutions: a survey", International Journal of Distributed and Parallel Systems (IJDPS) Vol.3, No.1, January 2012

[6] Zhu Qishen, Zhu Dongmei and Su Xunwen, "Distributed remote temperature monitoring and acquisition system based on CAN bus"; Prognotics and Health Management Conference, 2010, pp-1-4.

[7] "Photovoltaic pumping system based on Intel 80C196KC Microcontroller" IEEE 10th International Conference on Environment and Electrical Engineering, pp. 1-5, May 2011

[8] Md. Khairullah,Md. Habibur Rahman,S. M. Hasanul Banna, "BlueAd: A Location based Service using Bluetooth", International Journal of Computer Applications (0975 – 8887) Volume 43– No.15, April 2012

[9] Vinayak P. Musale,S. S. Apte, "Security Risks in Bluetooth Devices", International Journal of Computer Applications (0975 – 8887) Volume 51– No.1, August 2012

[10] Bruce Potter, Bluetooth security moves, Network Security, Volume 2006, Issue 3, March 2006, Pages 19-20, ISSN 1353-4858, http://dx.doi.org/10.1016/S1353-4858(06)70348-8.

[11] Heloise Pieterse, Martin S. Olivier, Bluetooth Command and Control channel, Computers & Security, Volume 45, September 2014, Pages 75-83, ISSN 0167-4048,

[12] Adam Laurie, Digital detective – Bluetooth, Digital Investigation, Volume 3, Issue 1, March 2006, Pages 17-19

[13] Bruce Potter, Warchalking and Bluejacking: Myth or reality, Network Security, Volume 2004, Issue 1, January 2004, Pages 4-5, ISSN 1353-4858

[14] Alexander M. Hainen, Stephen M. Remias, Darcy M. Bullock, Fred L. Mannering, A hazard-based analysis of airport security transit times, Journal of Air Transport Management, Volume 32, September 2013, Pages 32-38, ISSN 0969-6997

[15] G. Kabatianskii, B. Smeetsand, T. J ohansson, "On the cardinality of systematic A-codes via error correcting codes", IEEE Transact ion on Information Theory, vol. IT-42, pp. 566-578, 1996.

[16] Trishna Panse,Prashant Panse, "A Survey on Security Threats and Vulnerability attacks on Bluetooth Communication", International Journal of Computer

Science and Information Technologies, Vol. 4 (5) , 2013, 741-746

[17] Trishna Panse, Vivek Kapoor,"A Review on Security Mechanism of Bluetooth Communication", International Journal of Computer Science and Information Technologies, Vol. 3 (2) , 2012,3419-3422

[18] Pasquale Stirparo, Jan Löschner, "Secure Bluetooth for Trusted m-Commerce",Int. J. Communications, Network and System Sciences, 2013, 6, 277-288

[19] Tzu-Chang Yeh, Jian-Ren Peng, Sheng-Shih Wang, and Jun-Ping Hs,"Securing Bluetooth Communications", International Journal of Network Security, Vol.14, No.4, PP.229-235, July 2012

[20] K. Haataja and P. Toivanen, "Two practical man-in- the-middle attacks on Bluetooth secure simple pair-ing and countermeasures," IEEE Transactions onWireless Communications, vol. 9, no. 1, pp. 384-392, Jan. 2010.

[21] Heloise Pieterse, Martin S. Olivier, Bluetooth Command and Control channel, Computers & Security, Volume 45, September 2014, Pages 75-83, ISSN 0167-4048, http://dx.doi.org/10.1016/j.cose.2014.05.007

[22] Wei-Cheng Chu, Kuo-Feng Ssu, Location-free boundary detection in mobile wireless sensor networks with a distributed approach, Computer Networks, Volume 70, 9 September 2014, Pages 96-112, ISSN 1389-1286, http://dx.doi.org/10.1016/j.comnet.2014.05.005

[23] Manar Jammal, Taranpreet Singh, Abdallah Shami, Rasool Asal, Yiming Li, Software defined networking: State of the art and research challenges, Computer Networks, Volume 72, 29 October 2014, Pages 74-98, ISSN 1389-1286

[24] Marica Amadeo, Claudia Campolo, Antonella Molinaro, Giuseppe Ruggeri, Content-centric wireless networking: A survey, Computer Networks, Volume 72, 29 October 2014, Pages 1-13, ISSN 1389-128

[25] Evgeny Khorov, Andrey Lyakhov, Alexander Krotov, Andrey Guschin, A survey on IEEE 802.11ah: An enabling networking technology for smart cities, Computer Communications, Volume 58, 1 March 2015, Pages 53-69, ISSN 0140-3664

Authors' profiles

Soham Sengupta has 11 years of experience in teaching, research and industry. Since last five years he is channelizing his works and expertise in various interdisciplinary fields including but not restricted to the major domains of Computer and Electronics Engineering, to various development firms. He holds a Bachelor of Technology in Information Technology and the Masters in Mobile communication & Network Technology. His major expertise encompasses Computer Networks, Security, Wireless, GSM, and Augmented Reality, Java & Open sources, Scalable Rich Internet Applications, Distributed & Cloud Computing, Android plus cross-platform Mobile Application development etc.

Dr. Parthapratim Sarakr has been rendering his research expertise to teaching in DETS, University of Kalyani, India. Working there as the Senior Scientific Officer cum Professor, he has over 40 research journals encompassing microwave communication.

How to cite this paper: Soham Sengupta, Partha pratim Sarkar,"An Augmented Level of Security for Bluetooth Devices Controlled by Smart Phones and Ubiquitous Handheld Gadgets", IJIEEB, vol.7, no.4, pp.58-75, 2015. DOI: 10.5815/ijieeb.2015.04.08