Applying Packets Analysis as New Approach for Discovering Bluetooth Intrusion

Ghossoon. M. W. Al-Saadoon
Ass.Prof. Manager Center of D&R for Lectturrer Staff , Head Dep.of MIS,
College of Administrative Sciences, Applied Science University,
Kingdom of Bahrain ,Manama , Jufair ,P.O.Box:5055, Manama, Bahrain
Tel : +(973) 17728777- 149, Fax: +(973)17728915,
Ghowaleed2004@yahoo.com

Abstract- One of the main problems which are considered in this paper is **Bluesnarfing** which is a high jacking process that will take control over another Bluetooth device without permission. In this case the attacker is capable to tabulate and recognizes the victim's data such as phone book, calendar, messages and other features as well.

This paper resembles an analysis process on the Bluetooth protocol stack and an implementation on a security threat. A data analysis process to test and know how the live data were transferred on the protocol stack is captured. This analysis for the usage and roles of the layer such as HCI, RFCOMM, and L2CAP on the data captured.

A sniffing process will be done to monitor and sniffer software sniff the data transfer between two Bluetooth devices. The sniffing is done to analyze the data transferred and its properties. In this case, to fulfill the paper objective, a type of threat or attack will be **implemented to** enhance the core of a backdoor attack via Bluetooth protocol. Also **Security measures** have been developed to secure these cable connections so that information can travel safely. Now, as the time has passed, cables have become a nuisance. Bluetooth is one of the solutions to form a cable-free environment.

There is a second part of which is a **security threat were simulated** on Linux platform using Application Programming Interface programming language to generate an attack without permission which is also known as Bluesnarfing.

The implementation is to check the intrusion, the Bluesnarfing idea is carried out to apply it between two Bluetooth devices; it's either between two mobile phones or between laptop and mobile phone. Further, it is involved in research to provide more security on it to control threat or attack from any intruder regarding the protocol stack. The intrusion detection is the security part (intrusion) applying under Linux platform (Master) towards a slave (Nokia N82) to detect attacks.

***Keywords: Intrusion Security, BlueSnarfing, Bluetooth, Symbian Operating System mobile phone, and Monitoring Sniffer*.**

I.Introduction

In Bluetooth enabled environment, security is very crucial to maintain the reliability of the data transfer through the Bluetooth devices [2] . Since its creation, Bluetooth has transformed itself from a cable replacement technology to a wireless technology that connects people and machines. Bluetooth has been widely adopted on mobile phones and Personal Data Assistance (PDA). Many other vendors in other industries are integrating Bluetooth into their products. Although vendors are adapting to the technology, Bluetooth hasn't been a big hit among users.

This paper is mainly about the Bluetooth Data Analysis, threats detection and its implementation. The core of the paper is to perform an analysis of packets transfer between two Bluetooth devices using monitoring sniffer software of Bluetooth connection. Monitoring sniffer software is used to monitor the live packets captured between two Bluetooth enabled devices. It's crucial to understand the packet transfer and its properties together with its behavior for the further recording purposes. Security always was a major concern. Poor implementation of the Bluetooth architecture on mobile devices has led to some high profile Bluetooth hacks. Weak security protocol designs expose the Bluetooth system to some devastating protocol attacks.

II.Problems Statements

Bluetooth technology and associated devices are susceptible to general wireless networking threats, such as Denial of Service (DoS) attacks, eavesdropping, man-in-the-middle attacks, message modification, and resource misappropriation. They are also threatened by more specific Bluetooth-related attacks that target known vulnerabilities in Bluetooth implementations and specifications. Attacks against improperly secured Bluetooth implementations can provide attackers with unauthorized access to sensitive information and unauthorized usage of Bluetooth devices and other systems or networks to which the devices are connected. There are few types of problems that the Bluetooth technology might suffered from such as: Bluesnarfing, which is under focus in this paper. Bluejacking, Bluebugging, Car Whisperer [2], Denial of Service and Fuzzing Attacks.

III.Literature Review

In Bluetooth TM Security [7] proposed that Bluetooth is a technology that enables all kind of electronic devices to communicate with each other. Firstly one should note that a device's range normally is up to 10 meters, with a maximum of 100, thus the threats can be minimized in such a constraint environment by other means. Also the most common uses are for communication between mobile phones that normally do not require tremendous security features, whilst they do require low power consumption. This is also the reason why an LFSR based cipher was chosen instead of a well studied block cipher such as AES which is considered unbreakable today.

In Bluetooth Security Protocol Analysis and Improvements [1] which says that Bluetooth has transformed itself from a cable replacement technology to a wireless technology that connects people and machines. Bluetooth has been widely adapted on mobile phones and PDAs. Security remains a major concern. Poor implementation of the Bluetooth architecture on mobile devices has led to some high profile Bluetooth hacks. Weak security protocol designs expose the Bluetooth system to some devastating protocol attacks. This paper first explores four Bluetooth protocol-level attacks in order to get deeper insights into the weakness of the Bluetooth security design. We then propose enhancements to defend against those attacks. Performance comparisons are given based on the implementation of our enhancements on a software based Bluetooth simulator.

In Security Analysis of Bluetooth v2.1 + EDR Pairing Authentication Protocol [5], Bluetooth is designed for wireless communication between mobile devices. This paper provides a security analysis of the Bluetooth Version 2.1 + EDR pairing authentication protocol. An overview of how we modeled the security properties of Bluetooth using the Moor verification tool are provided, followed by our findings and analysis. Three different types of attacks were confirmed: rollback attacks, brute force attacks, and a denial of service attack.

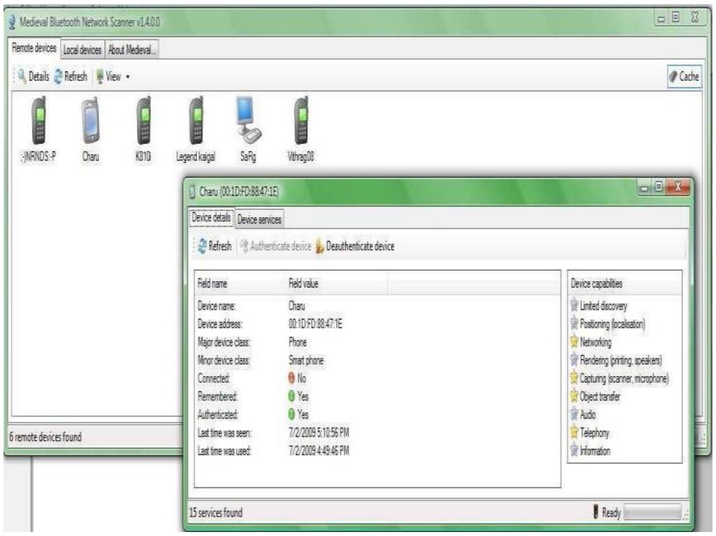
IV.Methodology

The main purpose of this paper is to sniff the Bluetooth protocol-level attacks in order to get deeper into the weakness of the Bluetooth security design. An analysis method is carried out to defend against those attacks. Performances of Bluetooth are given based on the implementation of enhancements on a software based Bluetooth simulator. Bluesnarfing allows hackers to gain access to data stored on a Bluetooth enabled phone using Bluetooth wireless technology without alerting the phone's user of the connection made to the device. The information that can be accessed in this manner includes the phonebook and associated images, calendar, and IMEI (International Mobile Equipment Identity).

A. System Design

In this system, it must be able to sniff data transferred among two Bluetooth enabled devices. Appropriate sniffer software is used to sniff the data transfer and its details accordingly. In the other hand, a type of threat is chosen to test on the security level of the protocol stack, and to know the weakness on the stack Bluesnarfing [3,6] enables attackers to gain access to a Bluetooth-enabled device by exploiting a firmware flaw in older devices. This attack forces a connection to a Bluetooth device, allowing access to data stored on the device and even the device's International Mobile Equipment Identity (IMEI). The IMEI is a unique identifier for each device that an attacker could potentially use to route all incoming calls from the user's device to the attacker's device.

Bluesnarfing is the unauthorized access of information from a wireless device through a Bluetooth connection, often between phones, desktops, laptops, and PDAs. This allows access to a calendar, contact list, emails and text messages and on some phones users can steal pictures and private videos. Currently available programs must allow connection and to be 'paired' to another phone to steal content. Bluesnarfing software is demonstrated and utilized weaknesses in the Bluetooth connection of some phones. This weakness has since been patched by the Bluetooth standard. There seems to be no available reports of phones being Bluesnarfed without pairing, because of the patching of the Bluetooth standard [4]. Bluesnarfing allows hackers to gain access to data stored on a Bluetooth enabled, as shown in Figure 1.



Field name	Field value
Device name:	Charu
Device address:	00:1D:FD:88:47:1E
Major device class:	Phone
Minor device class:	Smart phone
Connected:	● No
Remembered:	● Yes
Authenticated:	● Yes
Last time was seen:	7/2/2009 5:10:56 PM
Last time was used:	7/2/2009 4:49:46 PM

Figure 1: Remote Device

B. System Phases

In the system design, there are many software and hardware used to fulfill the requirement to achieve the objective of the paper, which is divided into three phases as follows:

1. *Phase 1*: Medieval Bluetooth Diagnostic Tools

Medieval Bluetooth Diagnostic Tool is one of the tools to sniff and monitor the data transfer between the two Bluetooth devices. Basically the nature of the tool is a service discovery tool, where it will discovers the devices around the range from the master device, besides give us the full services available for the particular devices. With this software it is able to analyze and scan local Bluetooth network .There are few the requirements to be satisfied to run this particular software as below:

1. An operating system of Microsoft Windows XP SP2 or Microsoft Windows Vista.
2. Microsoft .NET framework 2.0.
3. Hardware Bluetooth dongle installed on PC.

I. **Medieval Software** is to fully integrate Bluetooth technology into Windows platform, to achieve a smooth and transparent mobile phone and desktop PC interaction.

When client-server architecture was first born, the server and client computers was slow, cumbersome, and reserved to scientists, mathematicians, engineers, geeks and hackers; locked up in an institutional or academic environment. Today's mobile phones includes a lot of functionality, like: video and photo camera, audio and video player, videogames, GPS (Global Positioning System), PIM (Personal Information Manager), calendar, remote control and etc.

- **Devices Analysis Using Medieval Bluetooth Tools**

This paper will be conducted in two parts, which firstly is data sniffing procedures and the second one will be bluesnarfing implementation. In the first part, the service discovery procedures will be done at first then will be followed data transfer sniffer. The service discovery will be conducted using Medievel Bluetooth Tools which is Medievel Bluetooth File Transfer (OBEX FTP) for PC and Medievel Bluetooth Network Scanner.

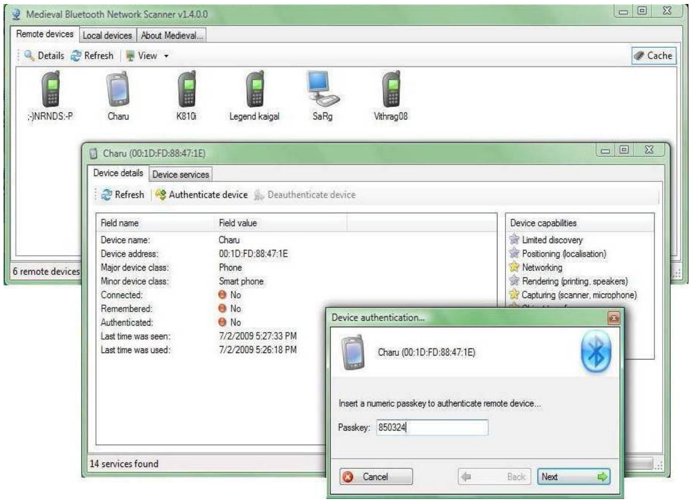
II. Medieval Bluetooth Network Scanner

Medieval Bluetooth Network Scanner can analyze and scan your Bluetooth network about the information of local and remote Bluetooth devices found. Software supports to browse supported services of each device in a clear and straightforward user-interface. A laptop with a Bluetooth dongle installed is needed and a Bluetooth enabled mobile is crucial to monitor the data transfer. The system requirement is Microsoft .NET Framework v2.0.

III. Scanning Algorithm

To establish this part of scanning for devices which Bluetooth enabled, as the first requirements are:

1. Turn on the Bluetooth in all the devices available in the range
2. Once it turn on, select the Refresh option.
3. Once the refresh completes it will shown all the devices that is available and hidden in the range. This is because the option cache is selected so that it can retrieve all the history of the available devices in the range.
4. For an example, try selecting a device in the range. A testing mobile is selected to see the service available. Note that even it’s not connected; the device properties and details are still available to be seen.
5. The devices service can be checked by clicking on the option device service. An authentication and authentication action can be performed for the selected device, an authentication and authentication on chosen device is to avoid its detail to be remembered, as shown in Figure 2.



Field name	Field value
Device name:	Charu
Device address:	00:1D:FD:88:47:1E
Major device class:	Phone
Minor device class:	Smart phone
Connected:	No
Remembered:	No
Authenticated:	No
Last time was seen:	7/2/2009 5:27:33 PM
Last time was used:	7/2/2009 5:26:18 PM

Charu (00:1D:FD:88:47:1E)

Insert a numeric passkey to authenticate remote device...

Passkey: 850324

Figure 2: Device Authentication

2. *Phase 2:* Wireshark Data Transfer Sniffer

Wireshark is an open-source packet sniffer computer application. It is used for troubleshooting network, analysis, software and communications protocol development, and for education. Wireshark provides the same function with Tcpdump, but it has graphical user interface so it is easier to use, and many more information and advance filtering options. It allows to see all traffic being transmitted all over the entire network (usually an Ethernet network but plug-in or other supports are being added for others) by putting the NIC (Network Interface Card) into the promiscuous mode. Wireshark or Ethereal uses the cross-platform GTK+ widget toolkit, and is cross-platform, running on variety computer operating systems like Linux, Mac OS X, and Windows.

I. WinPcap Requirements

Besides the requirements to install the Wireshark, it is also required to install supporting software such as WinPcap. Without WinPcap, it is not possible to capture live network traffic even if we are able to open saved captured files. WinPcap is a industry-standard tool for link-layer network access in Windows environments: it allows applications to capture and transmit network packets bypassing the protocol stack, and has additional useful features, including kernel-level packet filtering, a network statistics engine and support for remote packet capture.

WinPcap consists of a driver, which extends the operating system to provide low-level network access, and a library that is used to easily access the low-level network layers. This library also contains the Windows version of the well known libpcap Unix API.

WinPcap can be installed with normal installation processes which need to be done along with installation of the Wireshark. In fact, sometimes the latest version of WinPcap will come together with the Wireshark installer.

II. Algorithm of Data Analysis Using Wireshark

There are few steps to be done at first so that the Wireshark is applicable to sniff the wireless data in and out. The configuration will be as below:

1. Turn on the Bluetooth in the laptop (master) and the surrounding devices (slaves).

2. Then run a Bluetooth device discovery to find the available devices in the range, perform pairing with the favorable devices.

3. In this paper, the master is a personal laptop and the slaves are those shown in Figure 3.

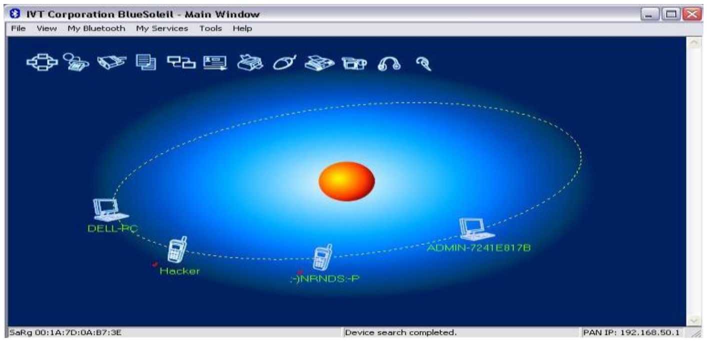


Figure 3: Bluetooth Device Pairing

To start the sniffing process of capturing the data transfer, it is a must to configure the interface of Wireshark and able to select our desirable interfaces according to the type of foundation that we are working on. The interface used is Bluetooth PAN Network NDIS Driver (Microsoft’s Packet

Scheduler) is used to sniff by using the Bluetooth interface and the data are captured accordingly.

1. To configure the selected interface, it can be done by selecting the option capture interface GUI.
2. Some important option can be configured as points below:

The interface of the sniffing process as the working is on Bluetooth personal area network, connecting few piconet and data transfer are tested within the piconet. That is why the option for link layer header type is choose as Ethernet where it is and internal sniffing.

The IP address of the sniffer and sniffed device. The IP for this process is not specified as the sniffing is not based on the IP address, but the protocol itself. If the IP address is specified, the Wireshark will only sniff any data from that particular IP address. Hence, in this case the IP address will be not specified.

III. Buffer Size

Enter the buffer size to be used while capturing. This is the size of the kernel buffer which will keep the captured packets, until written to disk. If it encounters packet drops, try increasing this value.

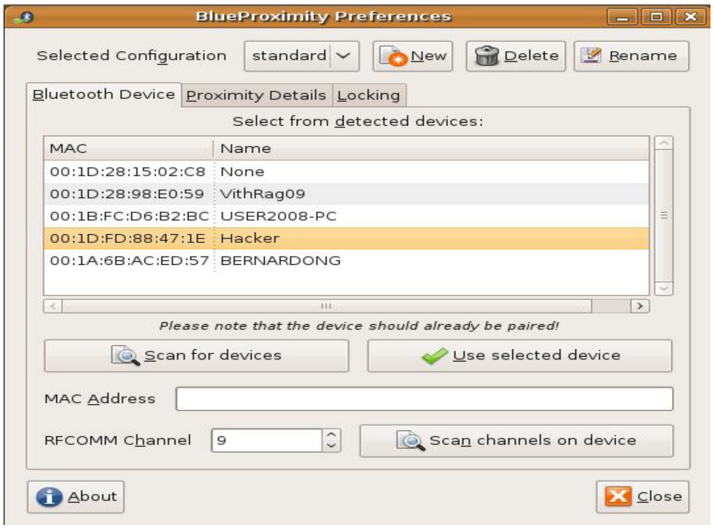
IV.Capture Filter

In this capture filter, the type of filter that we want to test the filtering must be selected. The specified the protocol filter that should be filtered, the IP never filter anything else other than the filtered protocol. Hence it is possible to specify any protocol in this case, but since the interfaces had been selected, it will be fine for leave it unselected. The sniffing process can be started to capture a live network data after setting are done

3. *Phase 3:* Bluesnarf **of Linux**

In this phase the implementation to Bluesnarf by the operation of Linux using Ubuntu version 8.1 as shown in the following algorithm:

I. Bluesnarfing USB algorithm implementation before executing the software, it is vital to set up the Bluetooth on the master device (Dell Inspiron 510m in this case). BlueProximity is capable to set up the computer to lock itself when the phone is out of bluetooth range, and unlock itself when it comes close enough again. Once the scanning process has finished, a device is selected as an automatic device so that it is set as an authorized device as shown in Figure 4.



BlueProximity Preferences

Selected Configuration: standard

Bluetooth Device

Select from detected devices:

MAC	Name
00:1D:28:15:02:C8	None
00:1D:28:98:E0:59	VithRag09
00:1B:FC:D6:B2:BC	USER2008-PC
00:1D:FD:88:47:1E	Hacker
00:1A:6B:AC:ED:57	BERNARDONG

Please note that the device should already be paired!

MAC Address:

RFCOMM Channel: 9

Figure 4: BlueProximity Preference

C. Paper Design

The paper design flow phases of the paper. Due to the requirement it is crucial to choose sniffing software to analyze packets. Sniffer software is chosen according to the environment and the type of the sniffing method, the sniffer will sniff the properties of the transferred data between two Bluetooth devices. In this case, the Wireshark Data Sniffer is used to capture live network data. In the second phase the program has to be written in order to implement the Bluesnarf between two Bluetooth devices. The program will be compiled and stimulated in under Linux platform. The source code is based on API Programming background. Throughout the analysis session, the validity of the program will be tested by performing stimulation. At the end of the paper, the results will be recorded and analyzed.

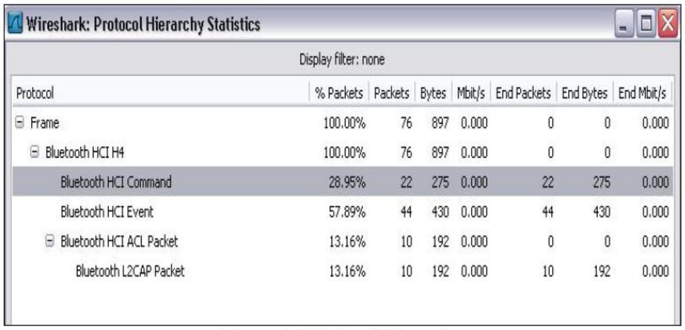
V.RESULTS

The results will be divided into two main parts: for the implementation of “Bluesnarf” and “Security”. The analysis will be conducted for each result as follows:

A. Bluesnarf Implementation

In this paper, the type of an attack will be implemented and simulated to analyze the protocol stack works on the usage as an attacker which is Bluesnarfing. In this process, the laptop will be executing the program to start the Bluesnarf process on the victim which is any mobile with the MAC address are being ping and being bluesnarfed.

- The analysis of Wireshark of data sniffing to transfer data between Bluetooth enabled devices are being tested by using Wireshark, and the output is obtained as below when a business card is sent from the master (laptop) to the slave (Nokia N82), as shown in Figure 5 .

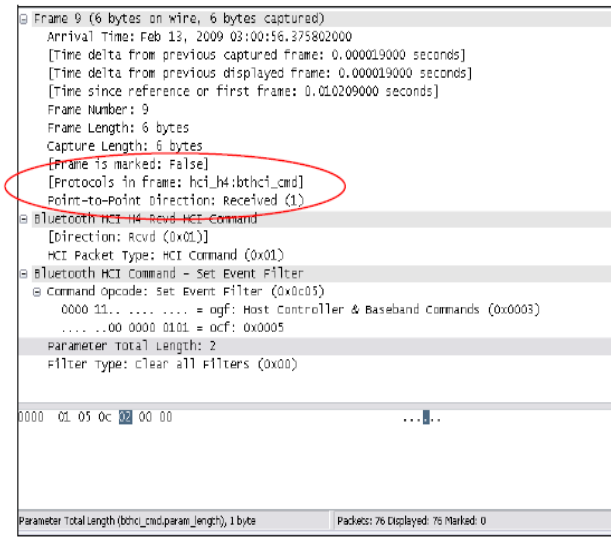


Protocol	% Packets	Packets	Bytes	Mbit/s	End Packets	End Bytes	End Mbit/s
⊟ Frame	100.00%	76	897	0.000	0	0	0.000
⊟ Bluetooth HCI H4	100.00%	76	897	0.000	0	0	0.000
Bluetooth HCI Command	28.95%	22	275	0.000	22	275	0.000
Bluetooth HCI Event	57.89%	44	430	0.000	44	430	0.000
⊟ Bluetooth HCI ACL Packet	13.16%	10	192	0.000	0	0	0.000
Bluetooth L2CAP Packet	13.16%	10	192	0.000	10	192	0.000

Figure 5: Wireshark Hierarchy Statistics

- Users should not accept transmission of any kind from unknown or suspicious devices. These types of transmission include messages, files and images. With the increase in the number of Bluetooth-enabled devices, it is important that users only establish connections with other rusted devices and only accept content from these devices.
- Ensure device mutual authentication is performed for all accesses; Mutual authentication is required to provide verification that all devices on the network are legitimate.
- Establish a “minimum key size” for any key negotiation process , establishing minimum key sizes ensures that all keys are long enough to be resistance to brute force attacks. Preferably, keys should be at least 128 bits long.
- Ensure that Bluetooth devices are turned off when they are not used, Bluetooth capabilities should be disabled on all Bluetooth devices, expects when the user explicitly enables Bluetooth to establish a connection. Shutting down Bluetooth devices when not is use minimizing exposure to potential malicious activities.
- Bluetooth HCI Command protocol is a command that allow the ACL link to be manipulated. Command packets used by the hosts to manage the controller and to monitor its status. They can be issued by ioctl (RSocket::Ioctl()) calls to a L2CAP or RFCOMM socket in this case. A HCI command packets contains the following :

1. An opcode identifying the type of command
2. Field giving total length in bytes of the following parameters
3. Parameters fields, as shown in Figure 6.

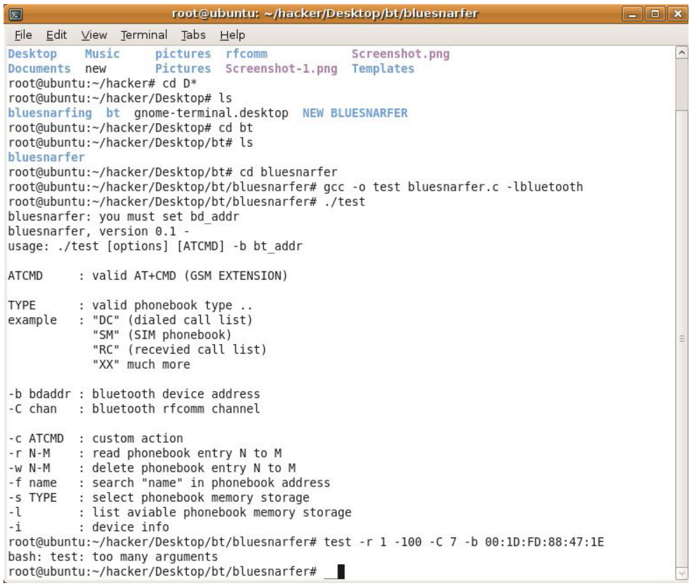


```
⊟ Frame 9 (6 bytes on wire, 6 bytes captured)
    Arrival Time: Feb 13, 2009 03:00:56.375802000
    [Time delta from previous captured frame: 0.000019000 seconds]
    [Time delta from previous displayed frame: 0.000019000 seconds]
    [Time since reference or first frame: 0.010209000 seconds]
    Frame Number: 9
    Frame Length: 6 bytes
    Capture Length: 6 bytes
    [Frame is marked: False]
    [Protocols in frame: hci_h4:bthci_cmd]
    Point-to-Point Direction: Received (1)
⊟ Bluetooth HCI H4 Rcvd HCI Command
    [Direction: Rcvd (0x01)]
    HCI Packet Type: HCI Command (0x01)
⊟ Bluetooth HCI Command - Set Event Filter
  ⊟ Command Opcode: Set Event Filter (0x0c05)
      0000 11.. .... .... = ogf: Host Controller & Baseband Commands (0x0003)
      .... ..00 0000 0101 = ocf: 0x0005
    Parameter Total Length: 2
    Filter Type: Clear all Filters (0x00)

0000  01 05 0c 02 00 00                                ......

Parameter Total Length (bthci_cmd.param_length), 1 byte    Packets: 76 Displayed: 76 Marked: 0
```

Figure 6: Bluetooth HCI Command

- The implementation of the actual bluesnarf The implementation of the actual bluesnarf operation in Linux using Ubuntu version 8.10, the important thing to set the details of locking setting and unlock settings as below:

1. Lock: Distance: 7 | time: 5 , It won't lock at once if bluetooth connection gets disrupted, something it seems to do for a second or two.
2. Unlock: Distance: 4, time: 1,Unlocks very fast when the phone get very close, as shown in Figure 7.


root@ubuntu: ~/hacker/Desktop/bt/bluesnarfer
File Edit View Terminal Tabs Help

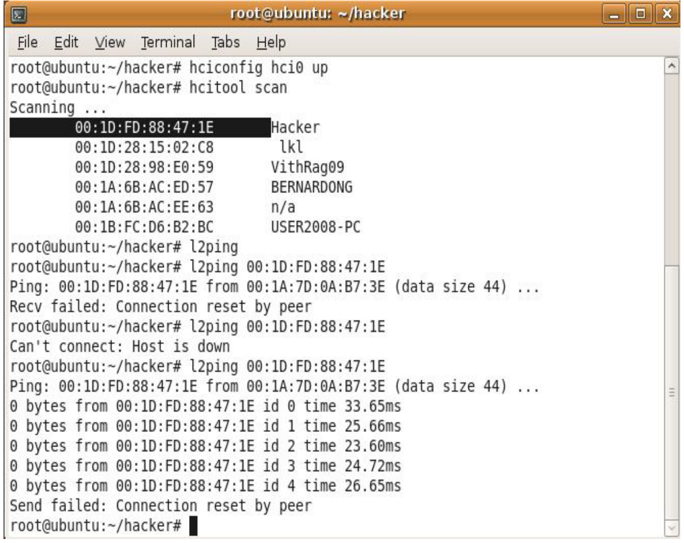
```
Desktop    Music     pictures  rfcomm            Screenshot.png
Documents  new       Pictures  Screenshot-1.png  Templates
root@ubuntu:~/hacker# cd D*
root@ubuntu:~/hacker/Desktop# ls
bluesnarfing  bt  gnome-terminal.desktop  NEW BLUESNARFER
root@ubuntu:~/hacker/Desktop# cd bt
root@ubuntu:~/hacker/Desktop/bt# ls
bluesnarfer
root@ubuntu:~/hacker/Desktop/bt# cd bluesnarfer
root@ubuntu:~/hacker/Desktop/bt/bluesnarfer# gcc -o test bluesnarfer.c -lbluetooth
root@ubuntu:~/hacker/Desktop/bt/bluesnarfer# ./test
bluesnarfer: you must set bd_addr
bluesnarfer, version 0.1 -
usage: ./test [options] [ATCMD] -b bt_addr

ATCMD     : valid AT+CMD (GSM EXTENSION)

TYPE      : valid phonebook type ..
example   : "DC" (dialed call list)
            "SM" (SIM phonebook)
            "RC" (recevied call list)
            "XX" much more

-b bdaddr : bluetooth device address
-C chan   : bluetooth rfcomm channel

-c ATCMD  : custom action
-r N-M    : read phonebook entry N to M
-w N-M    : delete phonebook entry N to M
-f name   : search "name" in phonebook address
-s TYPE   : select phonebook memory storage
-l        : list aviable phonebook memory storage
-i        : device info
root@ubuntu:~/hacker/Desktop/bt/bluesnarfer# test -r 1 -100 -C 7 -b 00:1D:FD:88:47:1E
bash: test: too many arguments
root@ubuntu:~/hacker/Desktop/bt/bluesnarfer# _
```

Figure 7 : Bluenarfing Output

B. Security Implementation

The security implementation has been carried out to enhance more security on the Bluetooth devices used in this paper.

The paper target device is Hacker. Hence the device is being ping from the master laptop to be checked between the master and slave, see Figure 8.



```
root@ubuntu:~/hacker# hciconfig hci0 up
root@ubuntu:~/hacker# hcitool scan
Scanning ...
        00:1D:FD:88:47:1E       Hacker
        00:1D:28:15:02:C8        lkl
        00:1D:28:98:E0:59       VithRag09
        00:1A:6B:AC:ED:57       BERNARDONG
        00:1A:6B:AC:EE:63       n/a
        00:1B:FC:D6:B2:BC       USER2008-PC
root@ubuntu:~/hacker# l2ping
root@ubuntu:~/hacker# l2ping 00:1D:FD:88:47:1E
Ping: 00:1D:FD:88:47:1E from 00:1A:7D:0A:B7:3E (data size 44) ...
Recv failed: Connection reset by peer
root@ubuntu:~/hacker# l2ping 00:1D:FD:88:47:1E
Can't connect: Host is down
root@ubuntu:~/hacker# l2ping 00:1D:FD:88:47:1E
Ping: 00:1D:FD:88:47:1E from 00:1A:7D:0A:B7:3E (data size 44) ...
0 bytes from 00:1D:FD:88:47:1E id 0 time 33.65ms
0 bytes from 00:1D:FD:88:47:1E id 1 time 25.66ms
0 bytes from 00:1D:FD:88:47:1E id 2 time 23.60ms
0 bytes from 00:1D:FD:88:47:1E id 3 time 24.72ms
0 bytes from 00:1D:FD:88:47:1E id 4 time 26.65ms
Send failed: Connection reset by peer
root@ubuntu:~/hacker# █
```

Figure 8: Ping Operations Between Master And Slave

The implementation is a simple one, but it is also a vital one so that there will be no backdoor attack and intruder hacking.

Also there was some basic security enhancement to make sure that the transfer will be more reliable and safe. The implemented ideas are listed as in Table 1.

Security Implementation	Security Need, Requirement, or Justification
Set Bluetooth devices to the lowest necessary and sufficient power level so that transmissions remain within the secure perimeter of the organization	**Setting Bluetooth devices to the lowest necessary and sufficient power level ensures a secure range of access to authorized users. Avoid using Class 1 devices because it have very extended range up to 100 meters**
Choose PIN codes that are sufficiently random and long. Avoid **static and weak PINs, such as all zeroes.**	**PIN codes should be random so that it can't be guessed and hacked. Longer PIN codes are more resistant to bruce force attack at least up to 8- character alphanumeric**
Ensure that link keys are based on combination key rather than unit keys	**The use of shared unit keys can lead to successful MITM attacks. The use of unit keys security was deprecated in Bluetooth v1.2**
Bluetooth devices should be configured by default as, and remain, undiscoverable expect as needed for pairing	**Bluetooth interfaces should be configured as non-discoverable, which prevents visibility to other Bluetooth devices expect when discovery is specifically needed. Also, the default self-identifying or discoverable names provided on Bluetooth devices should be changed to anonymous or any indentifying names so that there will be no aimed attack**
Service and profile lockdown of device Bluetooth stacks should be performed	**Many Bluetooth stacks are designed to support multiple profiles and associated services. The Bluetooth stack on a device should be locked down to ensure only approved profiles and services are** **available for use.**

Table 1: Security Implementation

VI.Discussion

This paper has been implemented and simulated under Linux and windows platform using sniffing methods.

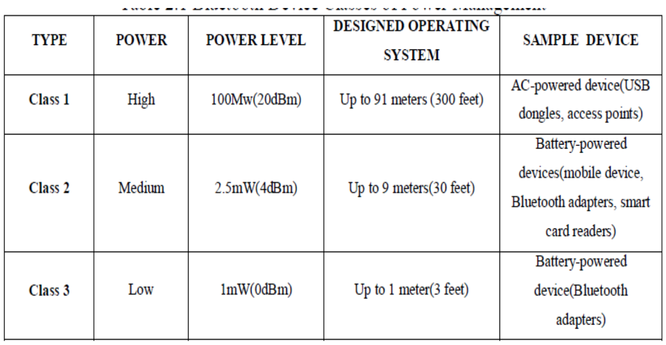
The protocol involved during the sniffing method and their percentage number of packets and the bytes values. The main protocol used in the Bluetooth stack is HCI protocol. The protocol Host Controller Interface (HCI) is used the most to transfer data via Bluetooth. HCI protocol provides a uniform interface method for accessing Bluetooth capabilities.

A network endpoint is the logical endpoint of separate protocol traffic of a specific protocol layer. The endpoint statistics of Wireshark will take the following endpoints into account:

1. **Ethernet:** an Ethernet endpoint is identical to the Ethernet's MAC address. In this case, each tab level shows the numbers of endpoints have been captured which is 4 Ethernet endpoint.
2. **IPv4:** an IP endpoint is identical to its IP address.
3. **UDP:** a UDP endpoint is a combination of the IP address and the UDP port used so different UDP ports on the same IP address UDP endpoint. In this case, there 9 UDP port were used.

The range of Bluetooth devices is characterized by three classes that define power management. Table 2 summarizes the classes, including their power levels in milliwatts (mW) and decibels referenced to one milliwatt (dBm), and their operating ranges in meters (m).4 Most small, battery-powered devices are Class 2, while Class 1 devices are typically USB dongles for desktop and laptop computers, as well as access points and other AC-powered devices.

Table 2: Bluetooth Device Classes of Power Management

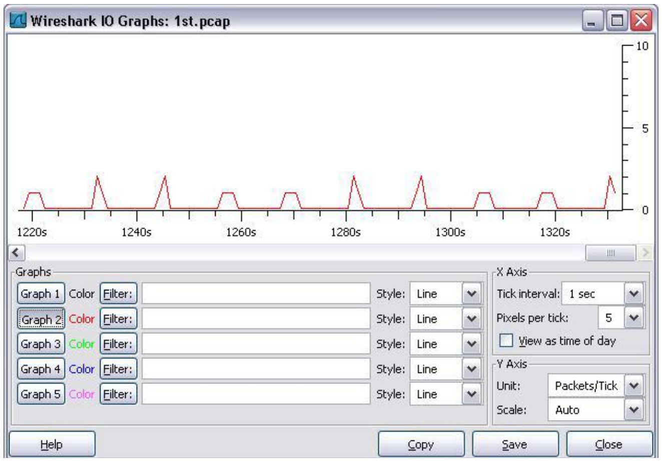


TYPE	POWER	POWER LEVEL	DESIGNED OPERATING SYSTEM	SAMPLE DEVICE
Class 1	High	100Mw(20dBm)	Up to 91 meters (300 feet)	AC-powered device(USB dongles, access points)
Class 2	Medium	2.5mW(4dBm)	Up to 9 meters(30 feet)	Battery-powered devices(mobile device, Bluetooth adapters, smart card readers)
Class 3	Low	1mW(0dBm)	Up to 1 meter(3 feet)	Battery-powered device(Bluetooth adapters)

So that Bluetooth devices can find and establish communication with each other, discoverable and connectable modes are specified. A device in discoverable mode periodically listens on an inquiry scan physical channel (based on a specific set of frequencies) and will respond to an inquiry on that channel with its device address, local clock, and other characteristics needed to page and subsequently connect to it. A device in connectable mode periodically listens on its page scan physical channel and will respond to a page on that channel to initiate a network connection. The frequencies associated with the page scan physical channel for a device are based on its Bluetooth device address. Therefore, knowing a device's address and the exact time are important for paging and subsequently connecting to the device.

VII.Conclusion

1. It can be concluded that the whole data captured can be converted into graph which indicates the packets transferred on y-axis versus time on x-axis. In this case, by average it's transferred about 2 packets or 200 bytes of data per tick, see Figure 9. The filter type can be chosen at any other graph column to make a comparison with another type protocol sniffing method. In this case, only the Bluetooth protocol will be sniffed even other filter is selected as the interface chosen was Bluetooth PAN Network Adapter Driver, hence it will only sniff the live data of Bluetooth protocol.


Wireshark IO Graphs: 1st.pcap

Graph 1 Color Filter: Style: Line
Graph 2 Color Filter: Style: Line
Graph 3 Color Filter: Style: Line
Graph 4 Color Filter: Style: Line
Graph 5 Color Filter: Style: Line

X Axis
Tick interval: 1 sec
Pixels per tick: 5
□ View as time of day

Y Axis
Unit: Packets/Tick
Scale: Auto

Figure 9: IO Graph

2. The conclusion for the analysis of the Wireshark Data Sniffing as the following:

The data transfer between Bluetooth enabled devices are being tested by using Wireshark, and the output is obtained as the following when a business card is sent from the master (laptop) to the slave (Nokia N82).

a. The Wireshark is capable in typically displaying information in three panels: the transmission overview, packet details and a pane showing raw data in hex value.

- The data can be analyzed according to 4 general statistic which are:
- Summary about the capture file.
- Protocol Hierarchy of the captured packets.
- Conversation between specific IP addresses.
- Endpoints of traffics to and from an IP addresses.

b. The protocol Host Controller Interface (HCI) is used the most to transfer data via Bluetooth.

c. HCI protocol provides a uniform interface method for accessing Bluetooth capabilities.

- Bluetooth HCI Command
- Bluetooth HCI Event
- Bluetooth HCI ACL Packets/ L2CAP Packets.

References

[1] Chi S. L., ”Bluetooth Security Protocol Analysis and Improvements”,Department of computer Science San Jose State University,May 2006.

[2]http://trifinite.org/trifinite_stuff_bluebug.html, copyright 2004-2006 trifinite .group.

[3] http://www.whatis.com.my.

[4] http://www.webopedia.com.my.

[5] Jersin John and Wheeler Jonathan, Security Analysis of Bluetooth v2.1 + EDR Pairing Authentication Protocol, Stanford University, March -25-2008.

[6] Karen S. & John P.” Guild to Bluetooth Security”, National Institute of standards and technology, September 2008 .

[7] Lu Y, W. Meier, and S. Vaudenay. The conditional correlation attack:A practical attack on bluetooth encryption. In Advances in Cryptology - Crypto 2005. Springer-Verlag, 2005. Available from http://www.iacr.org/conferences/crypto2005/p/16.pdf