

Hacking Bluetooth enabled mobile phones and beyond – **Full Disclosure**



Blackhat Security Briefings

April 1st 2005, Amsterdam, the Netherlands

Who we are

- Adam Laurie
 - CSO of The Bunker Secure Hosting Ltd.
 - Co-Maintainer of Apache-SSL
 - DEFCON Staff/Organiser
- Marcel Holtmann
 - Maintainer and core developer of the Linux Bluetooth Stack BlueZ
- Martin Herfurt
 - Security Researcher
 - Founder of trifinite.org

Bluetooth Technology Overview

- Bluetooth SIG
 - Trade Association
 - Founded 1998
 - Owns & Licenses IP
 - Individual membership free
 - Promoter members: Agere, Ericsson, IBM, Intel, Microsoft, Motorola, Nokia and Toshiba
 - Consumer <http://www.bluetooth.com>
 - Technical <http://www.bluetooth.org>

New device found: Airbus A310 Proceed with configuration ?



Bluetooth Technology

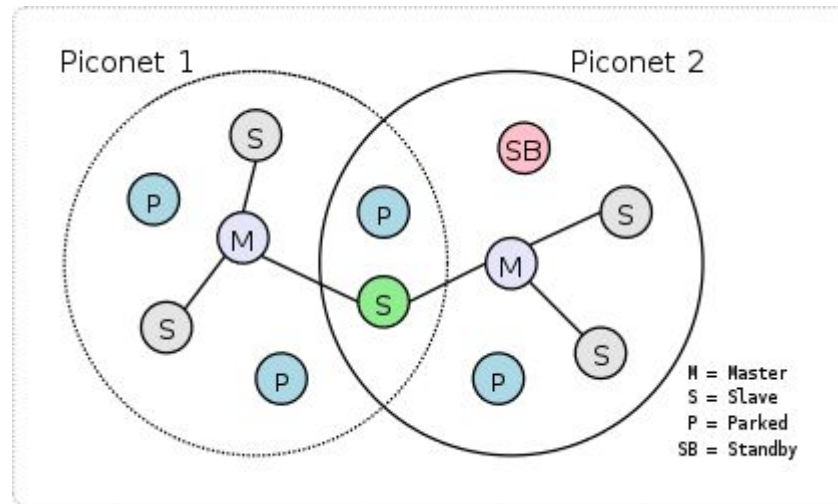
- Data and voice transmission
 - ACL data connections
 - SCO and eSCO voice channels
- Symmetric and asymmetric connections
- Frequency hopping
 - ISM band at 2.4 GHz
 - 79 channels
 - 1600 hops per second
 - Multi-Slot packets

Bluetooth Piconet

- Bluetooth devices create a piconet
 - One master per piconet
 - Up to seven active slaves
 - Over 200 passive members are possible
 - Master sets the hopping sequence
 - Transfer rates of 721 Kbit/sec
- Bluetooth 1.2 and EDR (aka 2.0)
 - Adaptive Frequency Hopping
 - Transfer rates up to 2.1 Mbit/sec

Bluetooth Scatternet

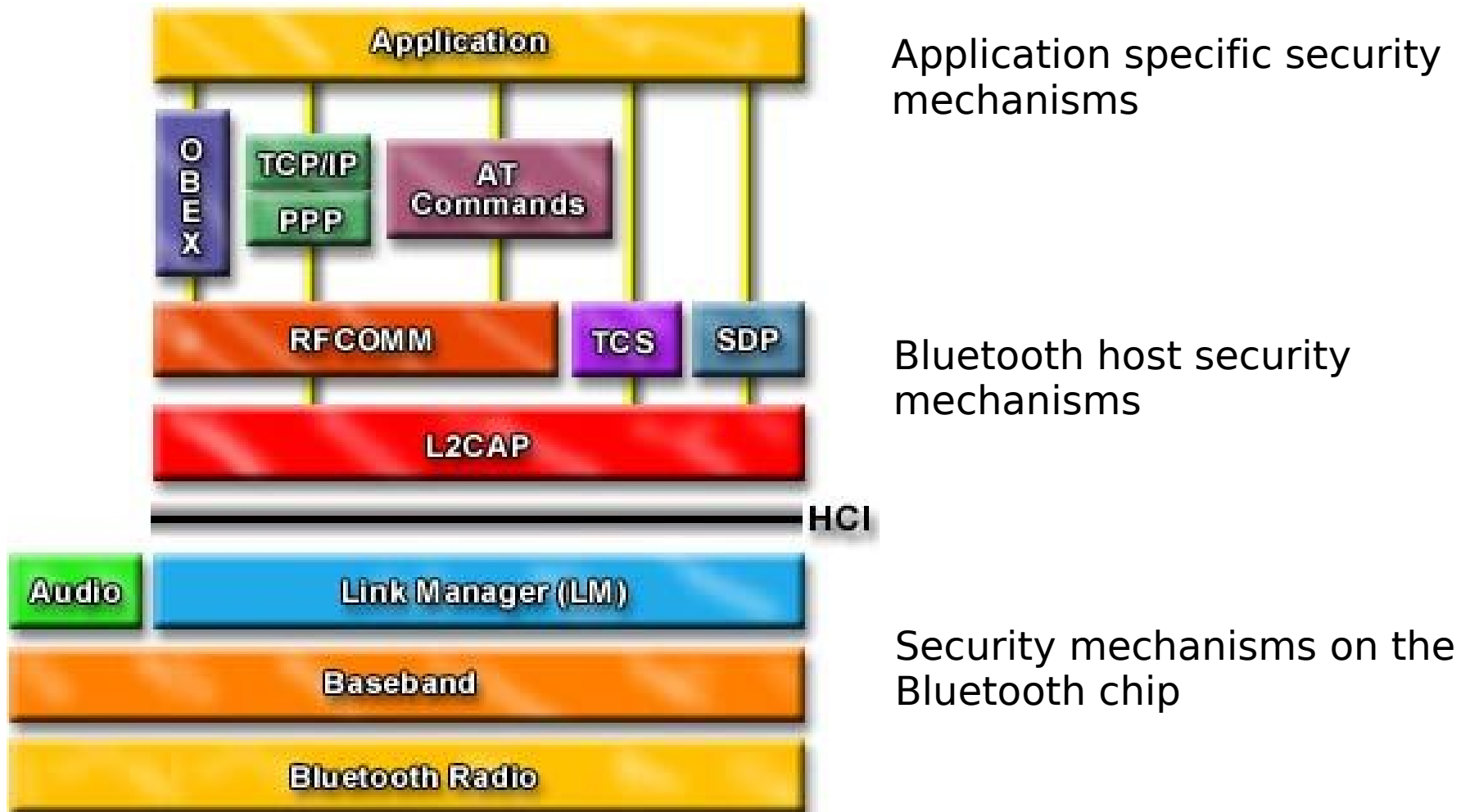
- Connected piconets create a scatternet
 - Master in one and slave in another piconet
 - Slave in two different piconets
 - Only master in one piconet
 - Scatternet support is optional



Bluetooth Architecture

- Hardware layer
 - Radio, Baseband and Link Manager
 - Access through Host Controller Interface
 - Hardware abstraction
 - Standards for USB and UART
- Host protocol stack
 - L2CAP, RFCOMM, BNEP, AVDTP etc.
- Profile implementations
 - Serial Port, Dialup, PAN, HID etc.

Bluetooth Stack



Bluetooth Security

- Link manager security
 - All security routines are inside the Bluetooth chip
 - Nothing is transmitted in “plain text”
- Host stack security
 - Interface for link manager security routines
 - Part of the HCI specification
 - Easy interface
 - No further encryption of pin codes or keys

Security Mode

- Security mode 1
 - No active security enforcement
- Security mode 2
 - Service level security
 - On device level no difference to mode 1
- Security mode 3
 - Device level security
 - Enforce security for every low-level connection

Linux and Bluetooth

```
# hciconfig -a
```

```
hci0:  Type: USB
      BD Address: 00:02:5B:A1:88:52 ACL MTU: 384:8  SCO MTU: 64:8
      UP RUNNING PSCAN ISCAN
      RX bytes:9765 acl:321 sco:0 events:425 errors:0
      TX bytes:8518 acl:222 sco:0 commands:75 errors:0
      Features: 0xff 0xff 0x8b 0xfe 0x9b 0xf9 0x00 0x80
      Packet type: DM1 DM3 DM5 DH1 DH3 DH5 HV1 HV2 HV3
      Link policy: RSWITCH HOLD SNIFF PARK
      Link mode: SLAVE ACCEPT
      Name: 'Casira BC3-MM'
      Class: 0x1e0100
      Service Classes: Networking, Rendering, Capturing, Object Transfer
      Device Class: Computer, Uncategorized
      HCI Ver: 1.2 (0x2) HCI Rev: 0x529 LMP Ver: 1.2 (0x2) LMP Subver: 0x529
      Manufacturer: Cambridge Silicon Radio (10)
```

```
# hcitool scan
```

```
Scanning ...
00:04:0E:21:06:FD      AVM BlueFRITZ! AP-DSL
00:01:EC:3A:45:86      HBH-10
00:04:76:63:72:4D      Aficio AP600N
00:A0:57:AD:22:0F      ELSA Vianect Blue ISDN
00:E0:03:04:6D:36      Nokia 6210
00:80:37:06:78:92      Ericsson T39m
00:06:C6:C4:08:27      Anycom LAN Access Point
```

Sniffing with hcidump

- Recording of HCI packets
 - Commands, events, ACL and SCO data packets
- Only for local connections
- Decoding of higher layer protocols
 - HCI and L2CAP
 - SDP, RFCOMM, BNEP, CMTP, HIDP, HCRP and AVDTP
 - OBEX and CAPI
- No sniffing of baseband or radio traffic

Security Commands

- HCI_Create_New_Unit_Key
 - HCI_{Read|Write}_Pin_Type
 - HCI_{Read|Write|Delete}_Stored_Link_Key
 - HCI_{Read|Write}_Authentication_Enable
 - HCI_{Read|Write}_Encryption_Mode
-
- HCI_Authentication_Requested
 - HCI_Set_Connection_Encryption
 - HCI_Change_Local_Link_Key
 - HCI_Master_Link_Key

Pairing Functions

- Events
 - HCI_Link_Key_Notification
 - HCI_Link_Key_Request
 - HCI_Pin_Code_Request
- Commands
 - HCI_Link_Key_Request_Reply
 - HCI_Link_Key_Request_Negative_Reply
 - HCI_Pin_Code_Request_Reply
 - HCI_Pin_Code_Request_Negative_Reply

How pairing works

- First connection
 - (1) HCI_Pin_Code_Request
 - (2) HCI_Pin_Code_Request_Reply
 - (3) HCI_Link_Key_Notification
- Further connections
 - (1) HCI_Link_Key_Request
 - (2) HCI_Link_Key_Request_Reply
 - (3) HCI_Link_Key_Notification (optional)

- Trivial OBEX PUSH channel attack
 - PULL known objects instead of PUSH
 - No authentication
- Infrared Data Association
 - IrMC (Specifications for Ir Mobile Communications)
 - e.g. telecom/pb.vcf
- Sony Ericsson T68, T68i, R520m, T610, Z1010
- Nokia 6310, 6310i, 8910, 8910i
- Devicelist on bluestumbler.org

- Trivial OBEX PUSH channel attack
 - Connect to Sync, FTP or BIP UUID/target
 - No authentication
 - Contents Browseable
 - Full read/write access
 - External Media Storage
- *new* for BlackHat
 - Number of devices affected not known

- Forced Re-keying
 - Authenticate for benign task (e.g. vCard exchange)
 - Force authentication if required (Mode 3)
- Partner deletes pairing
 - Hold connection open
 - Request Link Key Exchange
- Connect to unauthorised Channels
 - Serial Profile, OBEX FTP, etc.
- *new* for BlackHat

HeloMoto

- Requires entry in 'My Devices'
- OBEX PUSH to create entry
- Connect RFCOMM to Handsfree or Headset
 - No Key required
 - Full AT command set access
- Motorola V80, V5xx, V6xx and E398

BlueBug

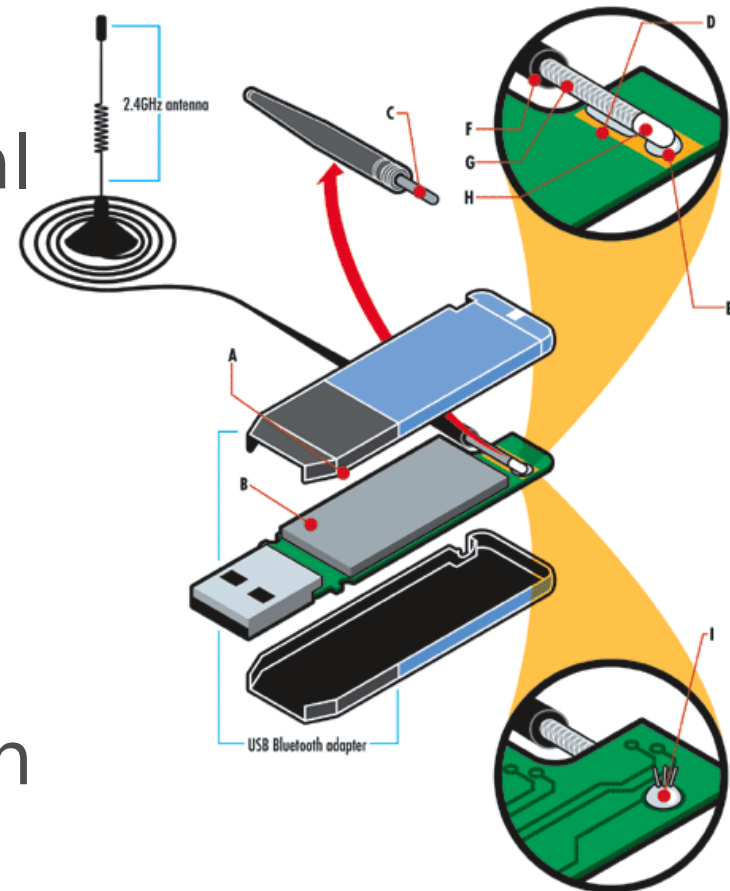


- Issuing AT Commands to covert service
 - BlueBug is based on AT Commands (ASCII Terminal)
 - Very common for the configuration and control of telecommunications devices
 - High level of control...
 - Call control (turning phone into a bug)
 - Sending/Reading/Deleting SMS
 - Reading/Writing Phonebook Entries
 - Setting Forwards
 - -> causing costs on the vulnerable phones!

Bluetooone



- Enhancing the range of a Bluetooth dongle by connecting a directional antenna -> as done in the Long Distance Attack
- Original idea from Mike Outmesguine (Author of Book: "Wi-Fi Toys")
- Step by Step instruction on trifinite.org



Long-Distance Attacking

- Beginning of August 2004
(right after DEFCON 12)
- Experiment in
Santa Monica California
- Modified Class-1 Dongle
Snarfing/Bugging Class-2
device (Nokia 6310i)
from a distance of
1,78 km (1.01 miles)



Bloover - What is it?



- Bloover - *Bluetooth* Wireless Technology Hoover
- Proof-of-Concept Application
- Educational Purposes only
- Phone Auditing Tool
- Running on Java
 - J2ME MIDP 2.0
 - Implemented JSR-82 (Bluetooth API)
 - Nokia 6600, Nokia 7610, Nokia 6670, ... Series 60
 - Siemens S65
 - SonyEricsson P900 ...



Bloover- What does it do? **Bloover**TM

- Bloover is performing the BlueBug attack
 - Reading phonebooks
 - Writing phonebook entries
 - Reading/decoding SMS stored on the device (buggy..)
 - Setting Call forward (predef. Number) +49 1337 7001
 - Initiating phone call (predef. Number) 0800 2848283
 - Not working well on Nokia phones :(but on some T610
- Please use this application responsible!
 - Not with phones of strangers...

Blueprinting – What is it? **Blueprinting**[™]

- Blueprinting is fingerprinting *Bluetooth* Wireless Technology interfaces of devices
- This work has been started by Collin R. Mulliner and Martin Herfurt
- Relevant to all kinds of applications
 - Security auditing
 - Device Statistics
 - Automated Application Distribution
- Released paper and tool at 21C3 in December 2004 in Berlin

Blueprinting - How



- Hashing Information from Profile Entries
 - RecordHandle
 - RFCOMM channel number
 - Adding it all up $(\text{RecHandle}_1 * \text{Channel}_1) + (\text{RecHandle}_2 * \text{Channel}_2) + \dots + (\text{RecHandle}_n * \text{Channel}_n)$
- Bluetooth Device Address
 - First three bytes refer to manufacturer
- Example of Blueprint

00:60:57@2621543

- Using L2CAP echo feature
 - Signal channel request/response
 - L2CAP signal MTU is unknown
 - No open L2CAP channel needed
- Buffer overflow
- Denial of service attack

Conclusions

- Bluetooth is a secure standard (per se)
 - Problems at application level
- Cooperation with Bluetooth SIG
 - Pre-release testing at UPF (UnplugFests)
 - Better communication channels for external testers
 - Security Expert Group mailing list
 - bluetooth.org more open areas
 - Mandatory security at application level

trifinite.org

- <http://trifinite.org/>
- Loose association of BT security experts
- Features
 - **trifinite.blog**
 - **trifinite.stuff**
 - **trifinite.album**
 - **trifinite.group**

trifinite.group

- Adam Laurie (the Bunker Secure Hosting)
- Marcel Holtmann (BlueZ)
- Collin Mulliner (mulliner.org)
- Tim Hurman (Pentest)
- Mark Rowe (Pentest)
- Martin Herfurt (trifinite.org)
- Spot (Sony)

Questions / Feedback / Answers

- Contact us via blackhat@trifinite.org
(group alias for Adam, Marcel and Martin)