More fun with Blue Radio Waves

Bluetooth sniffing is for free now!

by: iamabanana

TOC

- Introduction
- Bluetooth basics
- Sniffing Bluetooth
- BlueDrift – a bluetooth sniffer application
- Conclusions

Introduction

- Bluetooth sniffing is only possible with expensive special equipment.
- NO!
  - You only need a 20-30 euro Bluetooth adapter and a new firmware for it
- Here is the story...

The Story

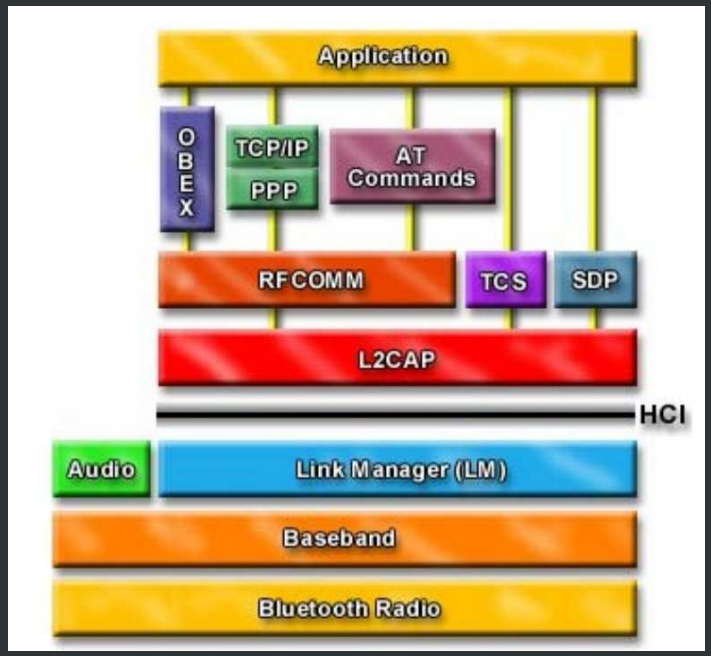
- Big hype about Bluetooth sniffing early this year
  - Max Moser – Busting the Bluetooth Myth
- Linux Bluetooth sniffer software released in August
  - Andrea Bittau – reverse engineered firmware to find commands to control sniffer
- Now we have the tools we need, to have even more fun with Bluetooth :)

Why Sniff Bluetooth?

- Because we can
- Security research
  - e.g. PIN cracking
- Non-security uses
  - Debugging your own applications
  - Reverse Engineering
    - The Wii remote (wiimote)
    - The PS3 controler
    - Other non-standard Bluetooth devices and protocols

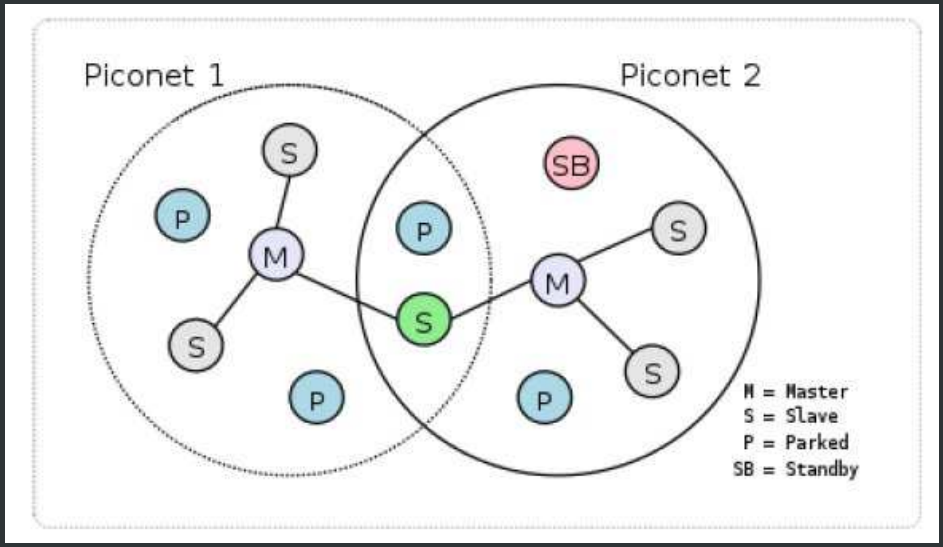
Bluetooth

- A general cable replacement technology
- Uses the 2.4GHz ISM band
- Defines protocol stack and application profiles



Bluetooth Network

- Master-Slave topology
  - Master can have up to seven active slaves
- Master defines the hopping sequence
  - 1600 hops per second accross 79 channels



Bluetooth Sniffing

- Local sniffing using hcidump
- Radio sniffing
- Air sniffing
- Some sniffing hardware:

Local Sniffing

- Sniff traffic of local computer using hcidump (see 1)
- Usefull for debugging

Radio Sniffing

- Sniffing using universal receiver (e.g. GNU Radio)
  - Sample entire Bluetooth frequency band
  - Do all protocols (including baseband) in software
- Advantages
  - Don't need to scan for devices
  - Sniff every connection around (nothing gets lost)
- Disadvantages
  - Very expensive and possible slow (no real time sniffing)

Air Sniffing

- Program a Bluetooth adapter to follow the hopping sequence of a specific device in order to sniff it's traffic
- Advantage
  - Cheap (no special hardware needed)
  - Real time sniffing
- Disadvantge
  - Need to know the BD ADDR of the target
  - One sniffer for each target

More on Air Sniffing

- Bluetooth uses frequency hopping
  - We can not just listen on a channel in order to sniff traffic
  - The hopping sequence is random (pseudo random)
    - Calculated from the BD ADDR and clock offset of the master

Practical Sniffing

- Need to know BD ADDR of piconet master device
  - We get the clock offset from the master
- Therefore the master needs to be visible
  - Could be a problem since the device that initiates the connection usually becomes the master, but the master can stay hidden since it only needs to see the slave
  - This could be solved by using GNU Radio to detect hidden Bluetooth devices (see 3)
  - Most of the time this should not be a problem

Building a Bluetooth Sniffer

- Requirements
  - Bluetooth adapter with **CSR BlueCore4-External** chipset
    - Check with: *hciconfig hciX revision*
  - Frontline sniffer firmware image
- Flashing
  - Change manufacturer ID of Bluetooth adapter to CSR
  - Flash adapter with: *dfutool hciX upgrade firmware.dfu*
  - Details in README from: *http://darkircop.org/bt/bt.tgz*

Hardware known to Work

- Cellink BTA-6030 (not available anymore)

- Fujitsu Siemens Bluetooth 2.0

- For more infos check:
  - http://www.evilgenius.de/2007/04/10/bluetooth-dongle-with-csr-chipset-and-flash-or-external-memory-using-flash/

Andrea Bittau's Frontline.c

- Basic Linux Bluetooth sniffer application
  - Configure Bluetooth sniffer hardware
    - Set BD ADDR of piconet master device
  - Read and dump traffic in hcidump format

Basic operation:

frontline -d hciX -f 7 (set filter to sniff all frames)

frontline -d hciX -S BD_ADDR@00:00:00:00:00:00 (set addr of master)

frontline -d hciX -e -w sniff.cap (sniff traffic and write to sniff.cap)

frontline -d hciX -s (stop sniffing)

hcidump -X -r sniff.cap (read and display capture file)

Frontline.c Limitations

- Sniffed packets don't have source or destination address, we only know if it is from master or slave
  - Can't tell BD ADDR of slave(s)
- Parser is not complete yet
  - Crashes quite often
- **The actual goal of Bittau is to write his own sniffer firmware so we can get rid of the frontline stuff!**

What can we do now?

- Bluetooth PIN cracking
  - http://openciphers.sourceforge.net/download/oc-v0.3.tgz
- Build our own sniffer application...

BlueDrift

- Bluetooth version of Driftnet
- Driftnet
  - Show images sniffed from TCP streams
- BlueDrift
  - Show files sniffed from OBEX transfers

OBEX

- OBejct EXchange (see 7)
  - Defined by Infrared Data Association
- Simple way to exchange arbitrary data
  - PUT file
  - GET file
- FTP mode
  - Create, delete and change directories
- Allmost all Bluetooth phones and PDAs support it

BlueDrift – how it works

- Device Scanner (1 Bluetooth adapter)
  - Scans for Bluetooth devices
- Sniffer
  - Sniff traffic of target devices
- File extractor
  - Dump plain file(s) from OBEX transfer in capture
- Viewer
  - Generate HTML page with filenames and content

The Sniffer

- We can only sniff the master
  - Sniffer needs to timeout if no traffic is seen in order to switch to another device (hopefully the master)
- BlueDrift supports multiple sniffers (tested with 2)
  - Better chance to catch actual transfer since the transfer could be over before we timeout and switch to next device
  - Sniff multiple piconets (transfers) at the same time

The File Extractor

- Patched version of hcidump (www.bluez.org)
  - Hardcoded some values to fit output of frontline.c, otherwise hcidump just doesn't parse
  - Added dump function to OBEX handler

The BlueDrift Framework

- Archives all capture files for later analysis
  - sniff_DATE+TIME_BDADDRofMASTER_hciX.cap
- Target selection can be changed easily
  - e.g. only sniff traffic from mobile phones or Nokia phones
- Takes care of some Bluetooth specialities
  - Don't scan while sniffing, since scanning can disturb transfers
- Written in Python

BlueDrift Attacks

- Sniff the high school (pr0n and violence)
  - http://www.heise.de/newsticker/meldung/94454
- Club/Disco - get the wannabie V.I.P.s
  - ..or some real Paris Hilton pictures ;-)
- Airports / Hotels / Trains
  - Can I give you my business card? Nop already have it.
- Cheat at Raffels and Rebates
- **I haven't tried any of these, it's your call!**

BlueDrift - Improvements

- Frontline.c is not completely stable...
- Fix hcidump or rewrite OBEX file extractor
  - Some captures crash hcidump
- Replace *Scanner* with *Trap* or *PassiveScanner*
  - **Trap** waits for incomming connections (e.g. SDP request) logs BD ADDR and hands it over to the sniffer
  - **PassiveScanner** uses GNU Radio to passively find BD ADDR of devices nearby (see 3)

Other cool applications...

- Bluetooth headset sniffer (listen to phone calls)
  - This is SCO traffic (have not tried this yet)
- Bluetooth TCP/IP sniffer
  - DUN (PPP over Bluetooth)
  - PAN (Ethernet over Bluetooth)
- Bluetooth HID (keyboard)
- **Most of these would require PIN cracking in order to decrypt the traffic!**

Conclusions

- Bluetooth sniffing now is *allmost* like wlan sniffing
  - New attacks will be found though this soon, I bet
- OBEX/OPP (object push profile) just got killed
- Need to change some concepts in Bluetooth
  - Allways encrypt traffic (e.g. use TLS/SSL) pairing doesn't work because this is more then encryption (you don't want to hand the key to your device to everyone you exchange files with)

Questions?

Links

1) http://www.bluez.org (Linux Bluetooth)
2) http://www.darkircop.org/bt/ (Linux Bluetooth Sniffer Application)
3) http://www.usenix.org/events/woot07/tech/full_papers/spill/spill_html/ (BlueSniff)
4) http://www.frontline.com (Manufecturer of Bluetooth Sniffers)
5) http://www.remote-exploite.org/research.html (Howto make a Bluetooth Sniffer)
6) http://www.trifinite.org (source of some slide material)
7) http://en.wikipedia.org/wiki/OBEX (OBEX overview)