Bluetooth — A Barn Door?

Thierry Zoller — Security Engineer

Goal of this Presentation

- Brief introduction to Bluetooth
  - History and origin
  - Function / Services
  - Difference vs. WiFi (802.11x)
- Risk potential for enterprises?
  - How to protect yourself
  - What needs to be considered
- What works, what doesn't — Reality Check
  - Risk transparency
- Live demos / BTCrack / Mac

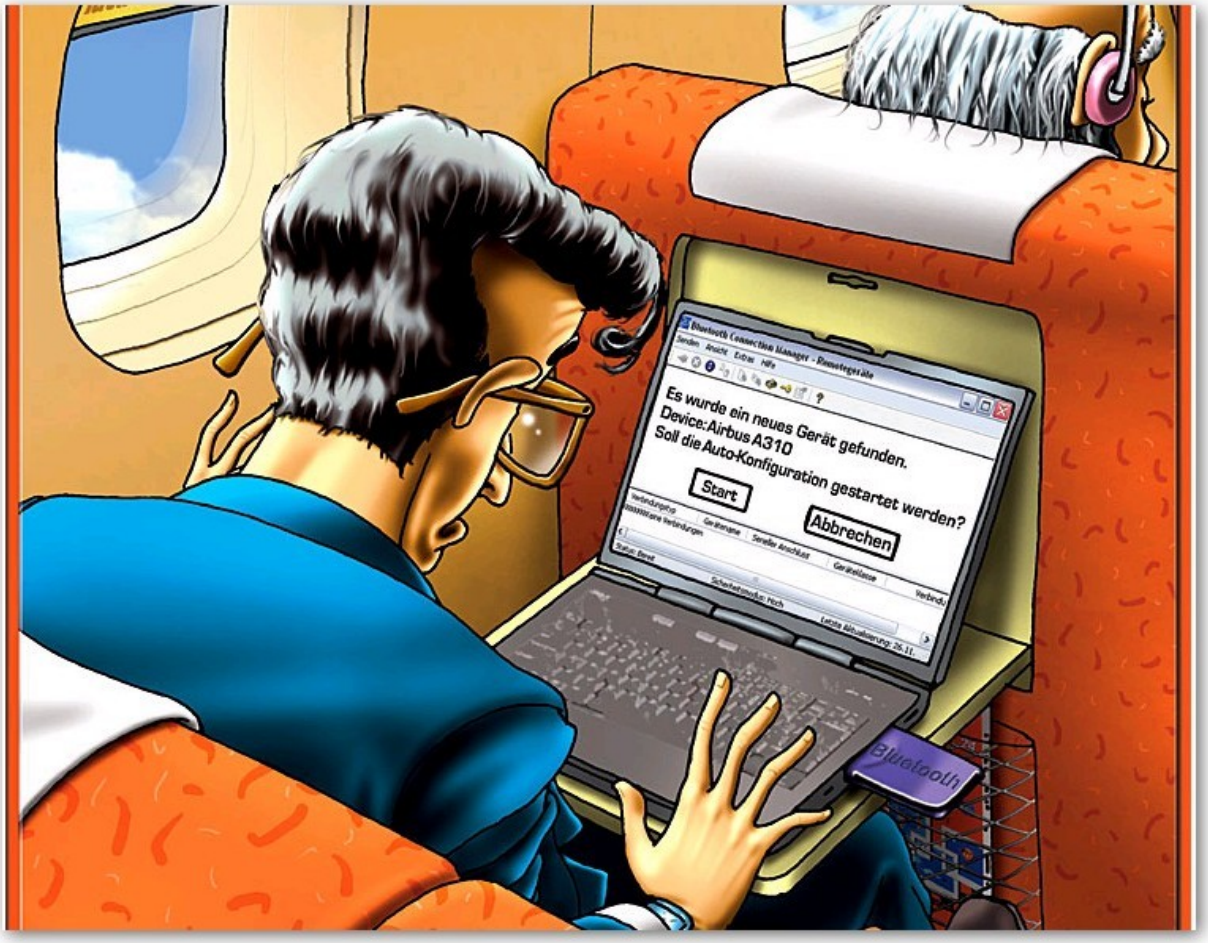
Who are we?

Kevin Finistere

- Former Head of Research at SNOSoft
- Vulnerabilities discovered in: Apple, IBM, SAP, Oracle, Symantec
- Contributed substantially to this talk

Thierry Zoller

- Security Consultant @ n.runs AG
- Vulnerabilities discovered in: Checkpoint, Symantec, Citrix VPN Appliance, MySQL, F-Secure, McAfee, NOD32, and 12 other AV vendors
- Don't enjoy talking about myself — Google it


Es wurde ein neues Gerät gefunden.
Device:Airbus A310
Soll die Auto-Konfiguration gestartet werden?
Start
Abbrechen
Bluetooth

Introduction to Bluetooth

What is Bluetooth (802.15)?

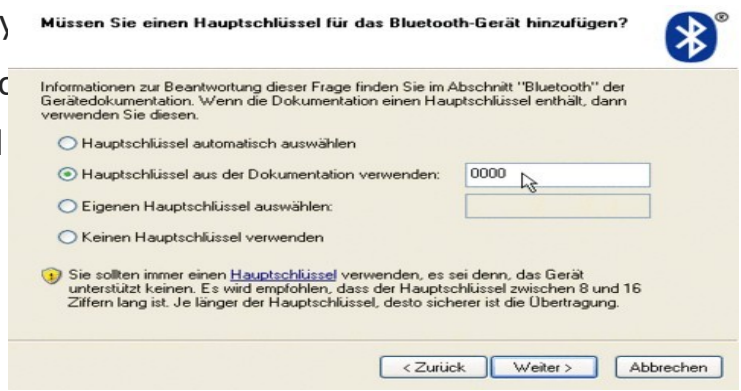
- Named after Harald Bluetooth
- Unregulated ISM band 2.4 GHz (2.400–2.4835 GHz)
- 79 channels (except France and Spain)
- Invented by Ericsson (1995) — SIG founded 1998 (Special Interest Group)
- Over one billion Bluetooth chipsets sold to date (as of 2006)
- **Goal: low-cost cable replacement, low power — PAN**

Difference vs. WiFi (802.11x)

- WiFi 11 channels | Bluetooth 79 channels
- WiFi SSID broadcast | Bluetooth passive
- Bluetooth "framework" (installed on every client)
- Frequency hopping
- Every Bluetooth participant is both AP and client
- Range

Bluetooth has 3 Security Modes

- Mode 1: device never enters security mode (no securit
- Mode 2: no encryption, security delegated to the applic
- Mode 3: link is encrypted before any data is exchanged
- Security Manager



What is Pairing?

- Two devices "pair" by authenticating to each other via PIN
- Keys (E21, E22) are exchanged, generated from device + user input
- Keys are stored, enabling later connections without re-entering the PIN
- After pairing, the device no longer needs to be "discoverable"

2 Pairing Modes

- **Non-Pairable Mode:**
  - Device rejects authentication handshakes (i.e. secured services are inaccessible)
- **Pairable Mode:**
  - Device responds with LMP_accepted and prompts for PIN entry

3 Discoverable Modes

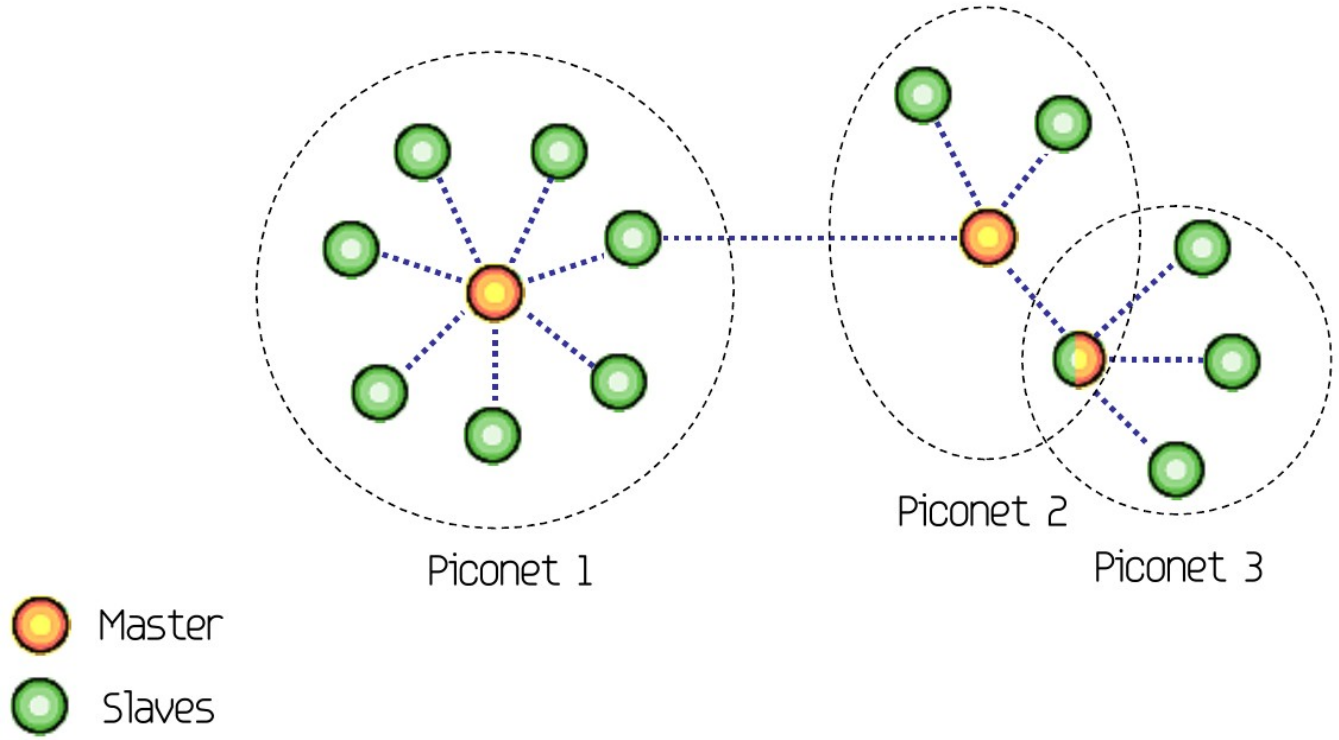
- **Discoverable Mode:**
  - Device can be found via an inquiry scan
- **Limited Discoverable Mode:**
  - Implementation-dependent: often time-based
    - Discoverable to devices on the Trusted list
    - Discoverable to devices on the Paired list
    - Phones are very often in Limited Discoverable mode just after power-on (airports…) without user confirmation. → BD_ADDR exposed
- **Non-Discoverable Mode:**
  - Does not respond to inquiry scans (in theory)

Bluetooth Addresses

- Bluetooth address "BD_ADDR" is a 48-bit MAC address
- Identifies devices like a MAC address on network gear
- First line of defence

00:11:9F:C5:F1:AE

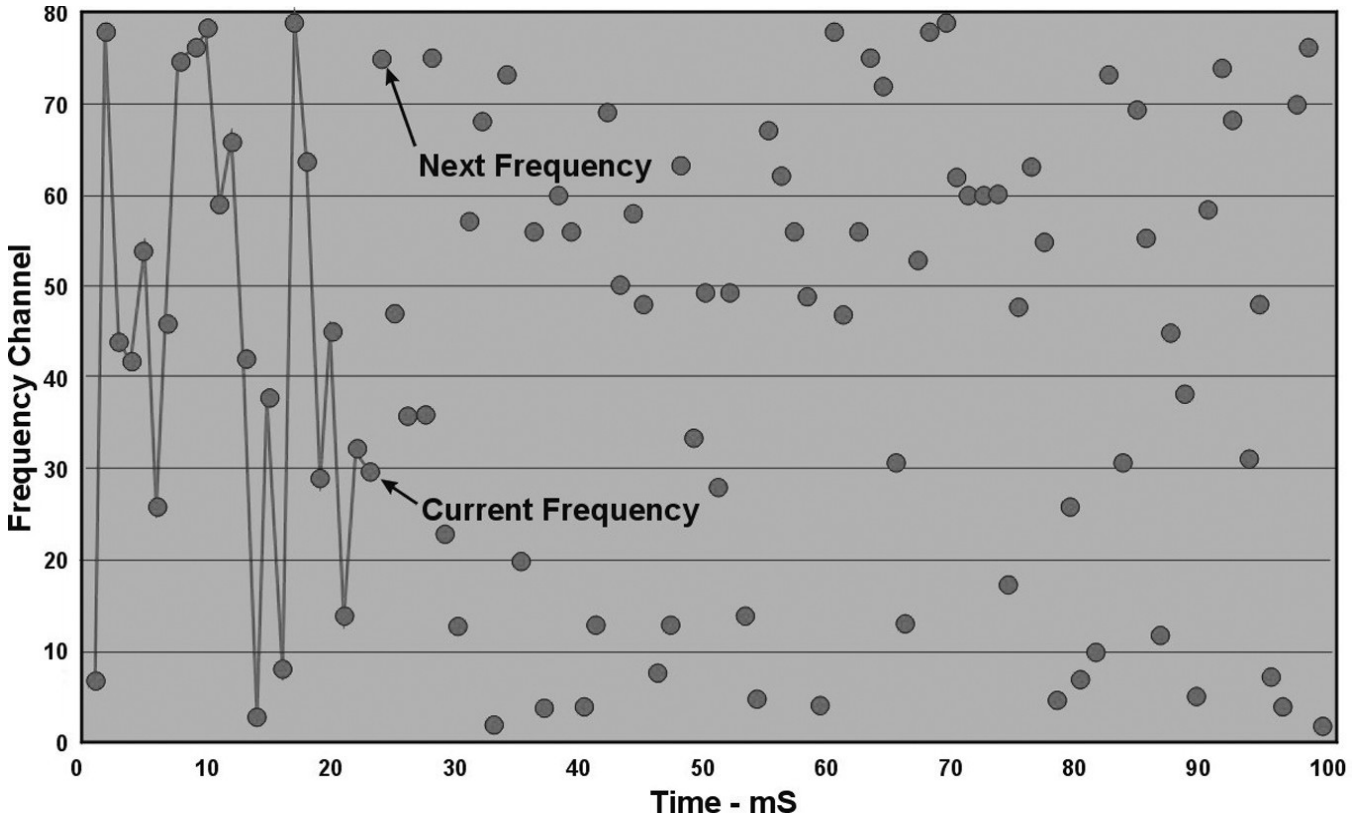
PAN — Personal Area Network



Bluetooth Frequency Hopping

- Slaves always synchronise to the master
  - Per spec, the master is the device that initiates paging
- Inquiry mode:
  - Master hops 3200 times/sec
  - Slave hops passively through preset frequencies
- Paging / Connected:
  - Piconet agrees on a hop sequence derived from the BD_ADDR and the master's clock-offset
  - Both hop synchronously 1600 times/sec
- *Passive sniffing is not trivial — frequency changes 1600x/sec. Only someone who knows the hop sequence can sniff or join.*

Bluetooth Frequency Hopping



Bluetooth Profiles

- Define services as a group of options and protocols offered
- Every Bluetooth device has them
- Vertical view of the BT stack (enumerated via SDP)
- e.g. Headset, Imaging, File Transfer, PIM, etc.

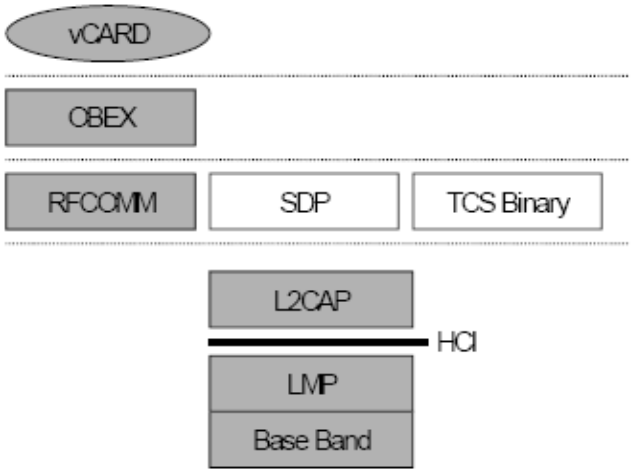
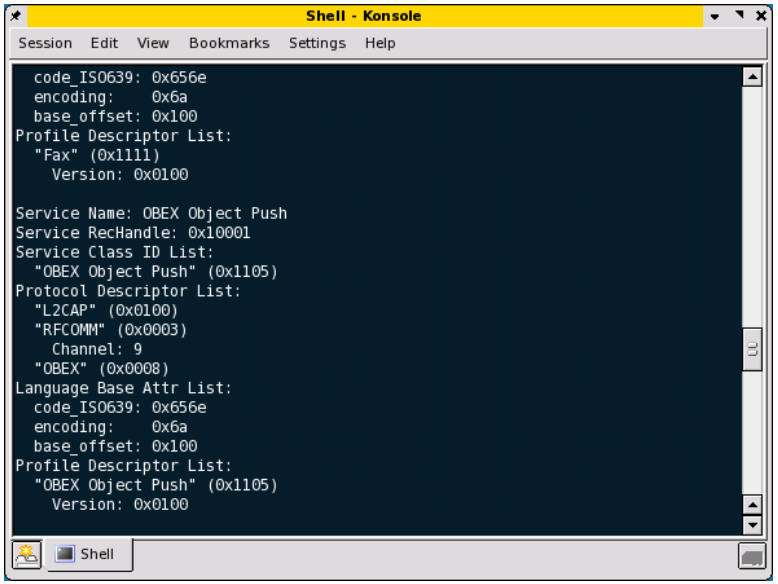


Figure 2 The Object Push Profile



Long-Range Bluetooth

Bluuuueeeeettooooooooooooooottttthhhhh

Long-Range Bluetooth

> Modify a commodity BT dongle so an external antenna can be attached. Instructions freely available online.

> Linksys is ideal

Class 3	10 m →	
Class 2	25 m →	
Class 1	100 m →	

Long-Range Bluetooth

- Antrum Lake (USA)
- 788 metres
- An old man "stole" the phone; Kevin tracked him with the Yagi.

Optimised Long-Range Bluetooth

- Integrated Linksys dongle
- Integrated USB cable
- Metallised directional antenna
- 10x optics
- Laser (coming soon)
- Higher penetration through walls

Experiment: found devices in the building behind the actual target building.

Automation

- Bundled
- Embedded Linux device (NSLU2)
- Autonomous scan & attack rig
  - Searches and archives results
  - Can attack devices automatically
  - Stores files and scans on SD card

Automation

It's Not Just Toys Using Bluetooth…

- Industry is jumping on:
  - Various pumps
  - Power utilities (UK) →

- *"The Bluetooth modems have been configured as non-discoverable […] the RL27 switches are protected from wireless hacking through a 48-bit software encryption key"*

- *"The operator can make software upgrades, reconfigure the RTUs, […] from a distance up to 100 metres."*

Bypassing Bluetooth Security

BlueBug Attack — Trifinite Group

- The service that should never have existed
  - Allegedly an IRDA leftover not intercepted by the Security Manager

BlueSnarf Attack — Trifinite Group

- "Get" request implementation over OBEX Push
- OBEX Push normally Security Mode 1

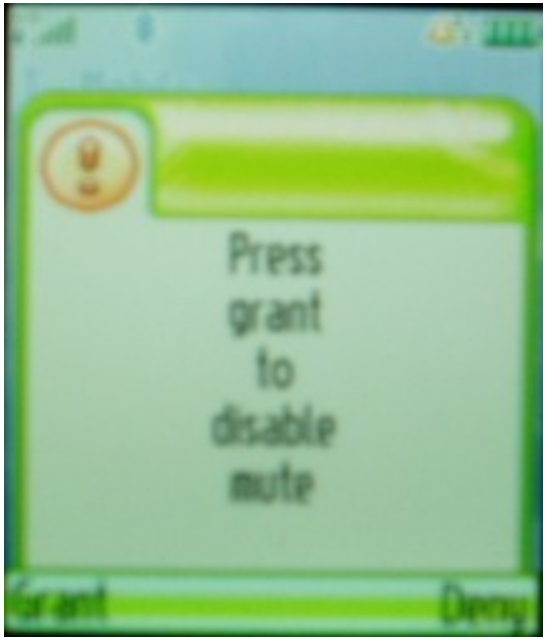
Give me your phonebook →

← OK

- Vulnerable devices:
  - Nokia 6310, 6310i, 8910, 8910i; Sony Ericsson T68, T68i, T610, R520m, Z1010, Z600; Motorola V80, V5xx, V6xx, E398; and others…

The "Please Press Yes" Attack

- Social engineering
- Motorola
- PEBL, V600, Razr, xxx?



Proof of Concept:

```
hciconfig hci0 name `perl -e
'print "Press\x0dgrant\x0dto\x0ddisable\x0dmute\x0d\x0d"'`
```

The PIN Isn't Really a PIN

> Did you know? Per the spec, the "PIN" is called a Passkey and may contain letters and even diacritics — not just digits.

> **BUT (almost) NO ONE HAS IMPLEMENTED IT THAT WAY**

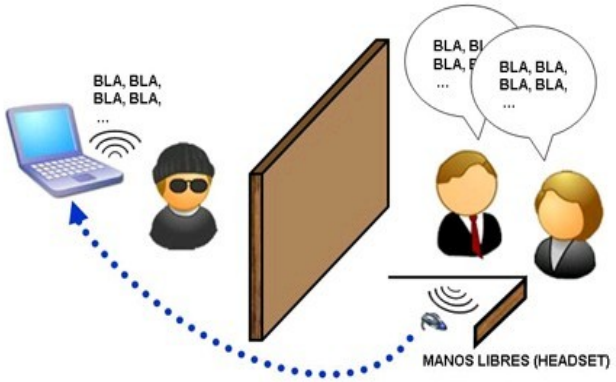
> Imagine if Microsoft had invented NTLM (after LM only allowed [a-z, A-Z, 0-9]) and then forgotten to let users type Umlauts.

> So brute-forcing a pairing exchange with BTCrack ALWAYS succeeds. Thanks. F-, see me after class.

Input	Bluetooth Internal
0123	0x30 0x31 0x32 0x33
Ärlich	0xC3 0x84 0x72 0x6c 0x69 0x63 0x68

"Eavesdropping"

- Root cause:
  - Discoverable mode
  - Pairing mode
  - Hard-coded PIN
- Laptops / PDA / Widcomm
- Real-time patch for CarWhisperer
- Windows native tooling




```
SWITCH: for ($bdaddr) {
  /00:02:EE/   && do { $pin="5475"; last;}; # Nokia
  /00:0E:9F/   && do { $pin="1234"; last;}; # Audi UHV
  /00:80:37/   && do { $pin="8761"; last;}; # O'Neill
  /00:0A:94/   && do { $pin="1234"; last;}; # Cellink
  /00:0C:84/   && do { $pin="1234"; last;}; # Eazix
  $pin="0000"; # 0000 is the default passkey in many cases
}
```

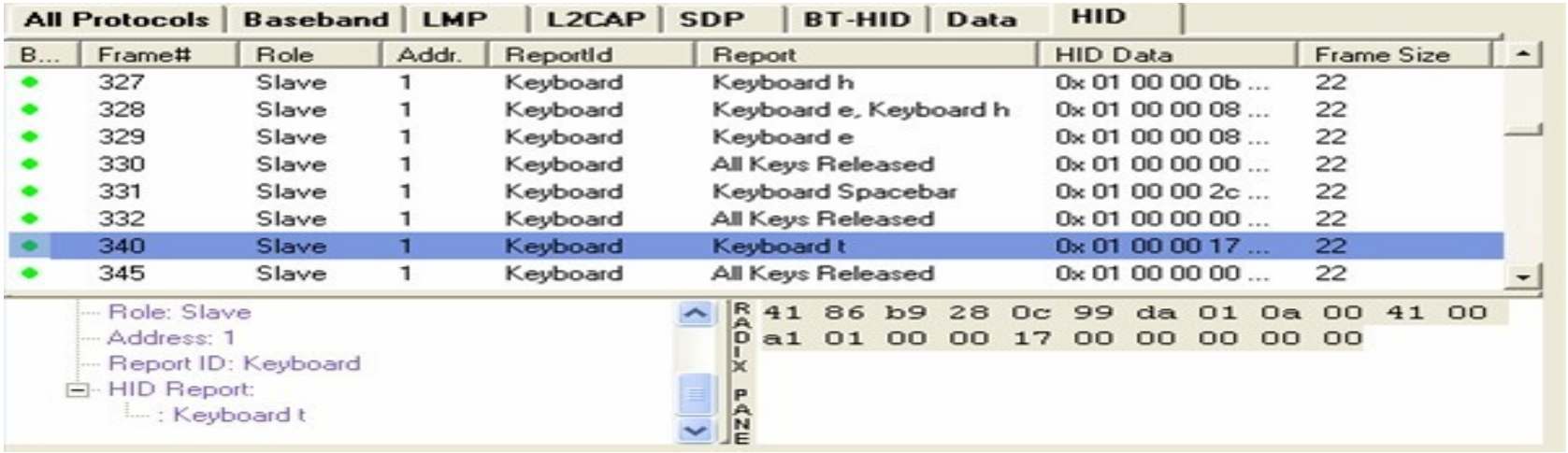
What the #!@#!! Is My Mouse Doing?

- HID = Human Interface Device
- Bluetooth service profile
  - HID server (PC)
  - HID client (keyboard & mouse)

- PSM channel 3 (PSM_Scan — Connect Success)
- Inject keystrokes (as if someone were sitting at the PC)
- Root cause:
  - Non-secure mode
  - Discoverable mode
- PoC: Collin Mulliner — HidAttack

What the #!@#!! Is My Mouse Doing?

- Encryption rarely used
- Sometimes no authentication (no Security Mode 3)
- Sniffing keystrokes
- Root cause:
  - Discoverable
  - Without authentication (or hardcoded PIN???)



Exploring Internal Networks (1)

- Or: "How does typing ../ even work here?"
- File systems freely accessible (!)
- Sometimes ROOT access (and onto the internal network, if the PC is wired)
- All known Windows drivers are or were affected
- Updates often impossible (Broadcom < Widcomm)
- Affected/still affecting: Windows, Mac, PocketPC, Linux, Unix

Example against a Pocket PC:

```
# ./ussp-push 00:11:B1:07:BE:A7@4
  trojan.exe
  ..\\..\\..\\..\\..\\windows\\startup\\trojan.exe
connected to server
Sending file: ..\..\..\..\..\windows\startup\trojan.exe,
  path: trojan.exe, size: 18009
Command (01) has now finished
```

Exploring Internal Networks (2)

- Mac OS X 10.3 & 10.4 Vanilla
- Patch issued a year ago
- OSX.Inqtana
- Remote root over Bluetooth

- Worked not only on Apple machines but also on Windows PCs for a long time
- Root cause:
  - ObexFTP → no authentication
  - Discoverable mode
  - Directory traversal (!)

Demo

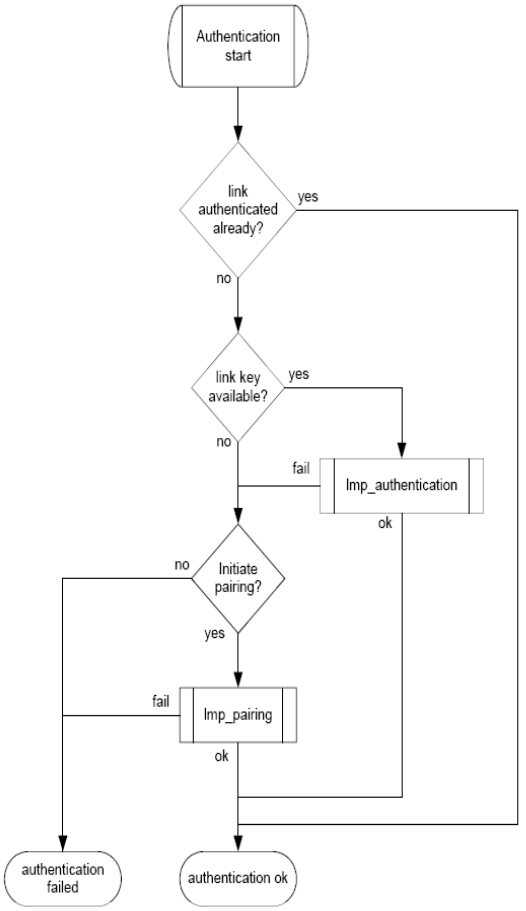
If it actually works…

Thanks to the burglar #!?

Bluetooth — A Barn Door? > Protocol issues

We Don't Care About the PIN

- The link key is more important from the attacker's perspective
- Why?
  - The link key alone is enough to authenticate
  - Only required when in pairing mode or discoverable mode
  - Encryption key (E0) is derived from the link key
  - One LK grants access to both devices
- Protocol 1.2 – 2.0 Authentication:



Finding Devices in Non-Discoverable Mode?

- Device must be actively communicating with another device
- PM_ADDR, AM_ADDR
- Target: BD_ADDR — 48-bit

00:11:9F:C5:F1:AE

- Set sniffer to a fixed frequency, sniff the preamble, extract the channel access code
- Read the error-correction field (baseband header — 10-bit CRC)
- First 8 bits assumed 00 (after OUI)
- Brute-force the remaining 8 bits (should be fast and reliable)

E0 Encryption

- "E0 is designed as a new cipher suite for Bluetooth"
  - "New cipher suite" = uh-oh

- **E0 should have a strength of 2^128**
- **E0 was reduced to 2^38 strength! (collisions)**

- E0 ↔ WEP: guarantees privacy but not security

- Checking whether E0 is in use? (see frequency hopping)
  - Often not the case at all
  - "Hardware" sniffers (FTS, BPA100, BPA105, Merlin)

- What strength? Spec mandates country-specific strengths.

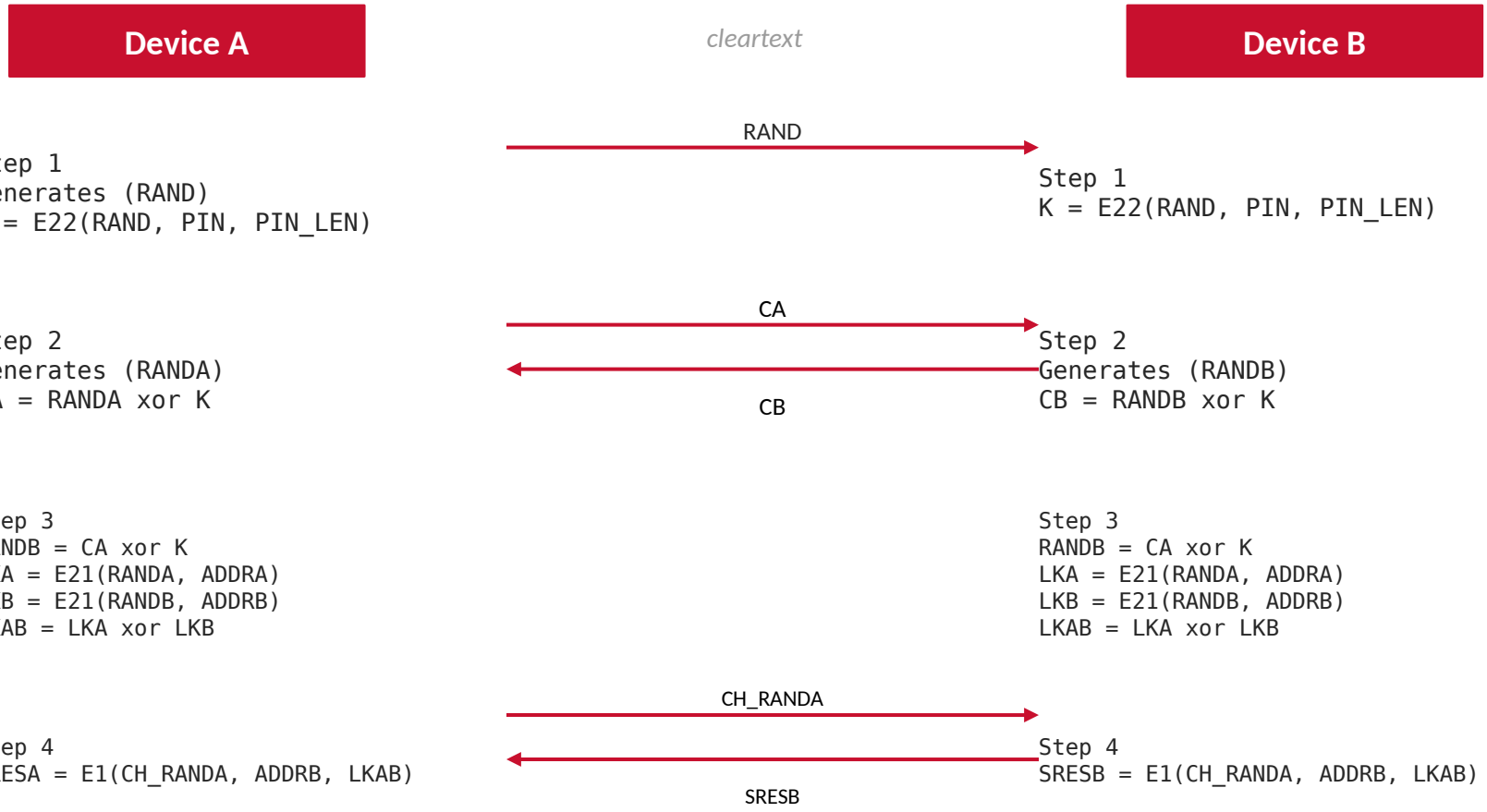
BTCrack — heisec Release

- What is BTCrack?
- What's new?
  - Minor bug fixes
  - Software speed: 185,000 → 200,000 keys/sec
- Collaboration with PICOComputing (David Hulton)
- FPGA support:
  - E12 @ 75 MHz = 10,000,000 keys/sec
  - E12 @ 50 MHz = 7,610,000 keys/sec
- SuperCluster = 15 FPGA boards
  - http://www.picocomputing.com

Results:

- 4-digit PIN: 0.035 s
- 5-digit PIN: 0.108 s
- 6-digit PIN: 4.312 s
- 8-digit PIN: 117 s (FPGA: 5.6 s)
- 9-digit PIN: 1318 s (FPGA: 101 s)

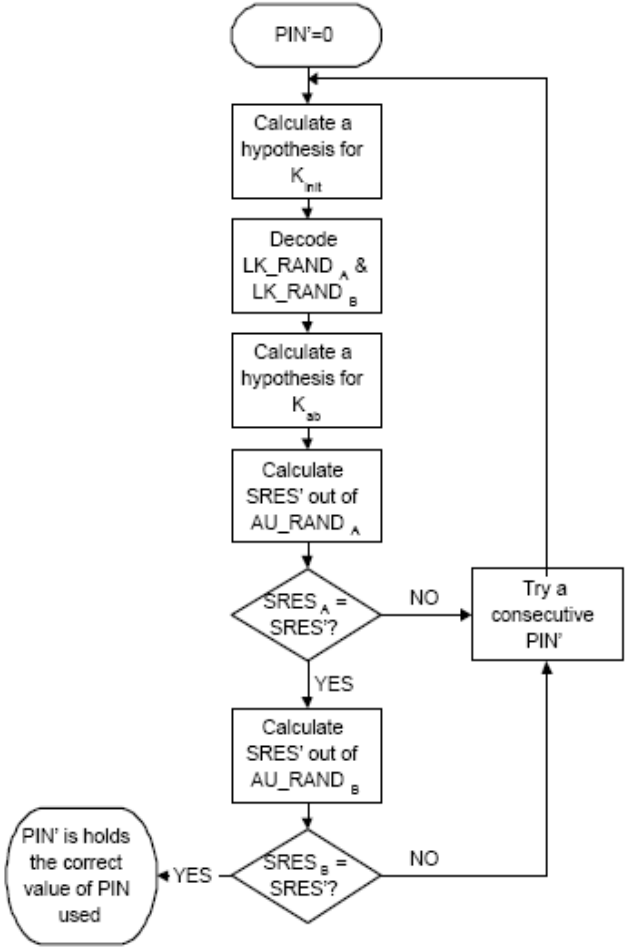
Pairing Handshake



BTCrack — heisec Release

```
Pin = -1;
Do
{
   PIN++;
   CR_K = E22(RAND, PIN, length(PIN));

   CR_RANDA = CA xor CR_K;
   CR_RANDB = CB xor CR_K;

   CR_LKA = E21(CR_RANDA, ADDRA);
   CR_LKB = E21(CR_RANDB, ADDRB);

   CR_LKAB = CR_LKA xor CR_LKB;

   CR_SRES = (CH_RAND, ADDRB, CR_LKAB);
}
while (CR_SRES == SRES)
```

Shaked & Wool logic:



Demo

But Don't I Need Expensive Hardware?

- Hardware sniffing was a myth — in reality almost everything runs on the software side
- What you need:
  - Dongle with CSR chipset and flash
  - e.g. CSR BC4 chipset
  - Firmware from a "three-letter" vendor
  - Linux tools: dfutool and bccmd
  - Google: "Busting the Bluetooth Myth"
  - Google: "Bluetooth Security seems to be very good compared"

- Since this method was disclosed by Max Moser, expect some novelties soon:
  - Layer 1–7 fuzzer
  - Open-source sniffer GUI
  - Improved hop-sync

Reality Check

> First-hand experience:

> Implementation flaws are always and easily exploitable (hardcoded PINs, BlueSnarf, etc.)

> Right now the pairing attack works reliably only under lab conditions. Reason: the commercial solution is poor and the range is short. With unlimited attempts it works in reality too. (We're working on it.)

> Devices in non-discoverable mode can be found (Redfang etc.) but it takes a lot of time and patience. Mobile target: barely feasible; stationary: feasible.

> Reconstructing the BD_ADDR with passive methods is feasible, but again limited by the meagre commercial gear. Yagi + flashed dongle + software works better.

> Simply "recording" conversations via headset is not as easy as claimed — unless you have the link key, and then you're back to the commercial-gear problem.

How Do We Protect Ourselves?

- Company policy → ban Bluetooth
- Vista Device GPO → Bluetooth USB dongles
- XP & XP SP2 → third-party software
  - Built-in chipsets

- If Bluetooth really must be present:
  - Update drivers — they are NOT updated through WSUS (unless using the Microsoft BT stack, since SP2). Push them.
  - Non-discoverable mode (and disable automatic scanning)
  - Harden BT services — set ALL to "Secure", disable unnecessary ones (Headset, Imaging, ...)
  - Even with services disabled, the client can still call them outbound (so information leakage cannot be eliminated this way)
  - Bluetooth 2.1

Summary

> Every Bluetooth-equipped device (PC, laptop) could inadvertently provide access into the internal LAN — and do so with native tooling (WiFi would require custom code, RCE aside)

> Bluetooth should be treated as a server protocol, not a client protocol — services are offered to the outside world and every one has potential weaknesses.

> Bluetooth bypasses your firewall, IDS, and other security infrastructure. Local firewalls don't protect the stack — I'm aware of no TDI drivers for Bluetooth that would.

> Bluetooth 2.1 shakes things up considerably on the security front:

- > Elliptic-curve cryptography
- > Near-Field Communication
- > Implementation flaws? (TO-DO)
- > Will it be any good?

Bluetooth — A Barn Door? > Kick out

Bluetooth — A Barn Door? > Kick out

Thierry Zoller

Solutions Consultant
Security

n.runs AG, Nassauer Straße 60, D-61440 Oberursel
phone +49 6171 699-0, fax +49 6171 699-199
Thierry.Zoller@nruns.com, www.nruns.com