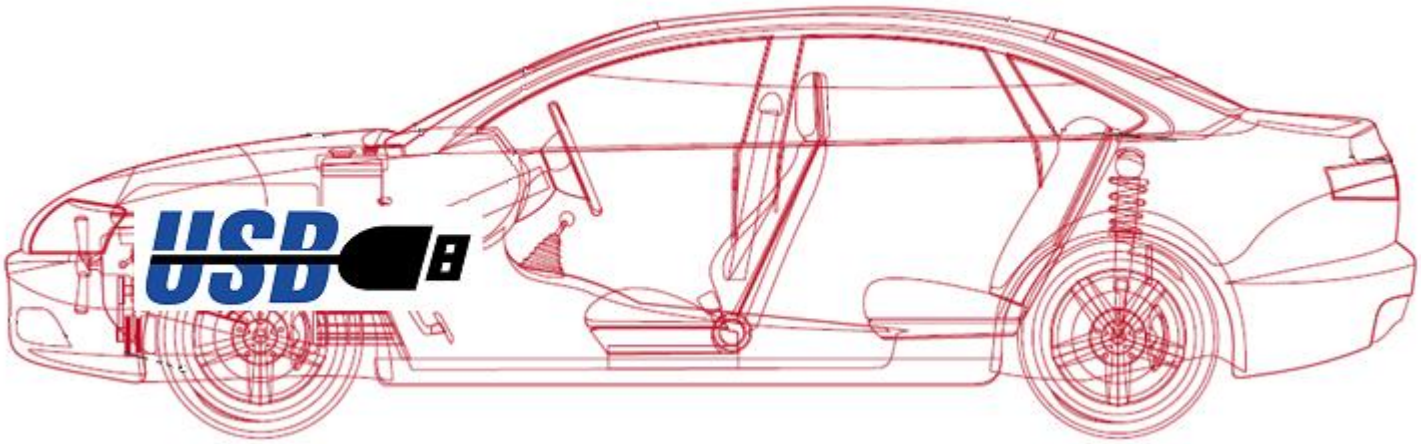
nccgroup
freedom from doubt

# USB under the bonnet

**Andy Davis, Research Director NCC Group**



# Who am I

- Research Director at NCC Group
- 10 years working in various security roles within UK Government
- 13 years working in commercial Cyber Security roles, primarily research-related
- Hands-on researcher - presented many security papers at conferences and identified over 100 vulnerabilities in USB hosts and devices.
- Responsible for all NCC Group's research output

nccgroup
freedom from doubt

# Agenda

- An overview of USB basics and some classic examples of where vulnerabilities have been previously identified
- How to test for USB host security vulnerabilities using a combined hardware/software approach
- The power of open source software when triaging bugs in embedded systems
- How exploitable are USB bugs?
- What are the unique challenges for USB security within an automotive environment?
- Lessons that can be learned from the non-embedded world

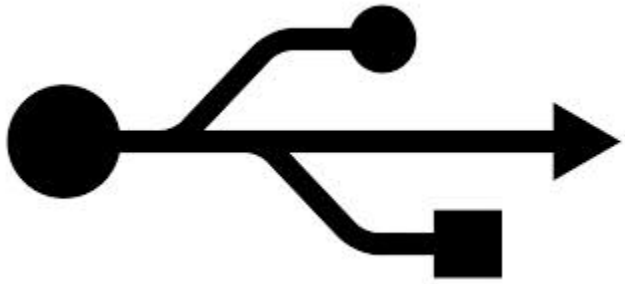
nccgroup
freedom from doubt

# Why talk about USB in cars?

- When did USB first appear in cars?
- What is USB used for in cars?
- Why is USB a security concern?
- How do attackers exploit USB vulnerabilities?

nccgroup
freedom from doubt

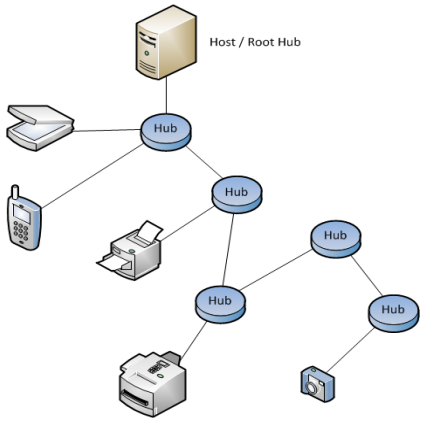
# USB Primer



# USB primer - architecture

- The aim of USB was to find a solution to the mixture of connection methods to the PC

- Currently four speeds of data transfer:
  - Low – 1.5Mbps (keyboard, mouse etc.)
  - Full – 12Mbps (originally for all other devices)
  - High – 480Mbps (developed in response to FireWire)
  - SuperSpeed – 5Gbps (Latest version – 3.x)




- Architecture is a tiered star topology:
  - Single host controller and up to 127 slave devices
  - A device can be plugged into a hub, and that hub can be plugged into another hub and so on. The maximum number of tiers permitted is six

- Host is Master - all communications on this bus are initiated by the host

- Devices cannot communicate directly with other devices (except for USB On-The-Go protocol)

# USB primer - terminology

nccgroup
freedom from doubt

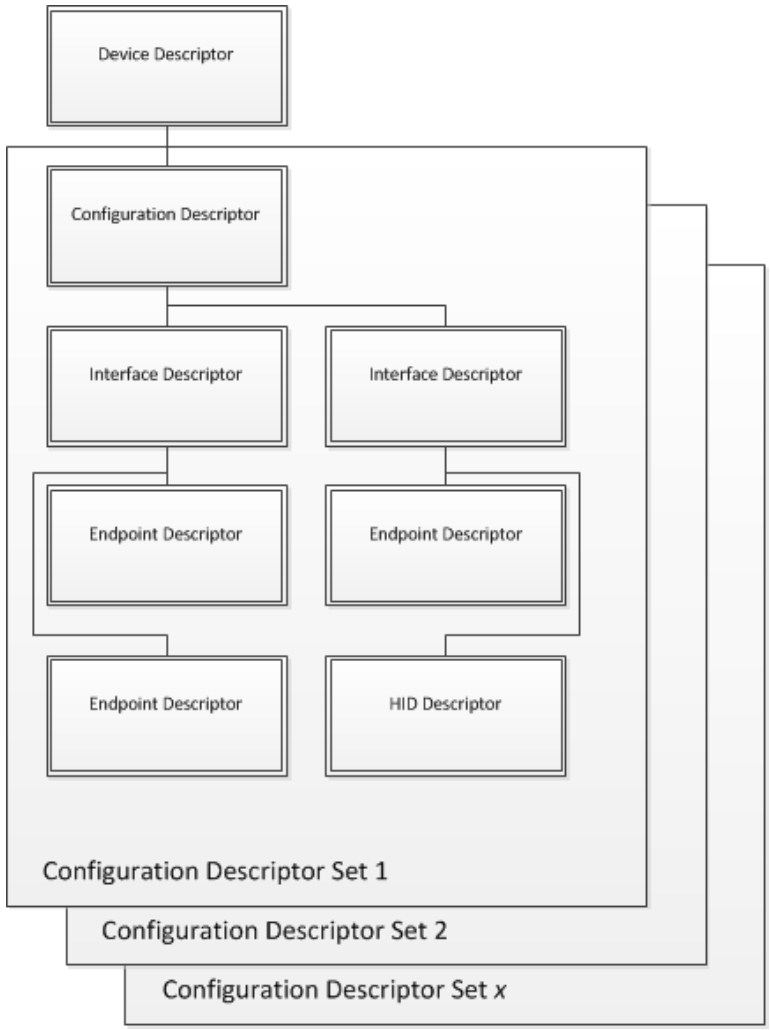
- The USB bus
  - When the host is transmitting a packet of data, it is sent to every device connected to an enabled port. It travels downwards via each hub in the chain which resynchronises the data transitions as it relays it. Only one device, the addressed one, actually accepts the data

- Endpoints
  - Each USB device has a number of endpoints. Each endpoint is a source or sink of data. A device can have up to 16 OUT and 16 IN endpoints.
  - OUT always means from host to device.
  - IN always means from device to host.
  - Endpoint 0 is a special case which is a combination of endpoint 0 OUT and endpoint 0 IN, and is used for controlling the device.

- Pipe
  - A logical data connection between the host and a particular endpoint, in which we ignore the lower level mechanisms for actually achieving the data transfers.

# Configurations, Interfaces, and Endpoints

- The device contains a number of **descriptors** (as shown to the right) which help to define the device's capabilities
- A device can have more than one **configuration**, though only one at a time, and to change configuration the whole device would have to stop functioning
- A device can have one or more interfaces. Each **interface** can have a number of endpoints and represents a functional unit belonging to a particular class
- Each **endpoint** is a source or sink of data



# USB enumeration

nccgroup
freedom from doubt

- **Pull-up resistor on data line** – Indicates device was connected (**reset device**)
- **Get Device descriptor** – what's the max packet size? – address 0
- **Reset then Set Address** - for the rest of the communications use this address
- **Get Device descriptor** – What are the device basic capabilities?

**HCI Driver**

- **Get Configuration descriptor** – What are the configuration details?
  - Interface descriptors
  - Endpoint descriptors
  - HID descriptors
- **Get String descriptors** – Get string language + product name etc.
- **Set Configuration** – Configuration is chosen – the device can be used

**USB Bus Driver**

- **Class-specific communication** - from this point onwards

**USB Device Driver**

# Recent USB host bugs in the IT world

- **CVE-2011-2295:** Oracle Sun Solaris USB Local Buffer Overflow Vulnerability
- **CVE-2012-3723:** Apple Mac OS X USB Hub Descriptor bNbrPorts Heap overflow
- **MS13-027:** The Windows 8 RNDIS kernel pool overflow
- **CVE-2013-3200:** Microsoft Windows USB Descriptor Handling Local Privilege Escalation

Image: http://www.biro-media.hr

# USB host bugs in IVI systems

- We have tested a number of different infotainment USB implementations
- Embedded systems are just as vulnerable to USB driver bugs as the IT world
- We have generally identified more bugs in embedded systems within vehicles than traditional operating systems
- Exploitation would only require physical access for seconds

Image: http://img.fruugo.com

nccgroup
freedom from doubt

# How I first started finding USB bugs (2011)

- Arduino microcontroller
- Fuzzer written in C++
- Only emulates USB HID devices
- Only allows semi-automated fuzzing
- Identified bugs in:
  - Windows 7
  - Windows XP
  - OS X
- Limitations – not really fast enough to emulate most USB devices

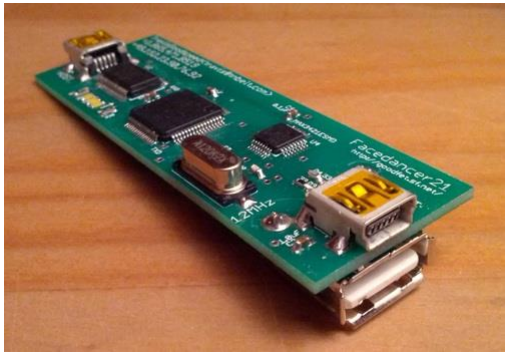
# USB fuzzer – the next generation (2012)

- Dedicated USB test equipment hardware
- USB capture and playback
- Emulates any USB host or device
- Understands and analyses the different USB device classes
- Uses a scripting language to generate USB traffic
- Costs approx. USD1200 (plus specific class analysis options)
- Limitations – doesn't have a software API to control it


Image: www.mqp.com

# How I find them now (2013 onwards)

- Facedancer and umap



```
$ sudo python umap.py -P /dev/ttyUSB0 -i

----------------------------------------
 _   _ _ __ ___   __ _ _ __
| | | | '_ ` _ \ / _` | '_ \
| |_| | | | | | | (_| | |_) |
 \__,_|_| |_| |_|\__,_| .__/
                      |_|

The USB host assessment tool
Andy Davis, NCC Group 2013
Version: 1.0

Based on Facedancer by Travis Goodspeed

For help type: umap.py -h
----------------------------------------

01:01:00 - Audio : Audio control : PR Protocol undefined
01:02:00 - Audio : Audio streaming : PR Protocol undefined
02:02:01 - CDC Control : Abstract Control Model : AT commands V.250
02:03:ff - CDC Control : Telephone Control Model : Vendor specific
02:06:00 - CDC Control : Ethernet Networking Control Model : No class-specific protocol required
03:00:00 - Human Interface Device : No subclass : None
**SUPPORTED**
06:01:01 - Image : Still image capture device : Bulk-only protocol
07:01:02 - Printer : Default : Bidirectional interface
08:06:50 - Mass Storage : SCSI : BBB
 **SUPPORTED**
09:00:00 - Hub : Default : Default
 **SUPPORTED**
0a:00:00 - CDC Data : Default : Default
0b:00:00 - Smart Card : Default : Default
```

- Umap is open source software - https://github.com/nccgroup/umap
- Facedancer is open source hardware - http://goodfet.sourceforge.net/hardware/facedancer21/

Image: http://blog.j-michel.org/

nccgroup
freedom from doubt

# Demo: USB testing

# The power of open source

- Fuzz testing is very effective, but only identifies *potential* bugs
- To triage bugs, either a development environment is required…
- …or if the USB stack is open source you can triage on Linux


Image: http://images.atelier.net

# How else can USB be exploited?

- Malicious firmware updates
- Malicious content that attacks media parsers
- Viral code that uses USB as the transport mechanism
- USB-based vendor engineering tools usage

Image: http://us.cdn4.123rf.com

# USB exploitation restrictions

- Bug triage is often challenging
- Driver exploit development is not for the faint hearted!
- USB descriptors often have limited space for attack payloads
- Race condition bugs can sometimes have strict timing limitations
- Physical access is required in order to insert a malicious USB device
- But…


Image: http://pagesunforgotten.files.wordpress.com

# USB exploitation impact

- **Code Execution:** Running malicious code on an infotainment system, potentially (depending on vehicle architecture) resulting in access to the CAN bus and to cyber physical systems


- **Data Loss:** Exfiltrating sensitive data such as Personally Identifiable Information (PII) or credit card details from apps running on the infotainment system

- **Malware:** Installing malware that stays resident within the vehicle and performs undesirable actions

Image: http://www.smartenergyjournal.com

# Unique challenges in automotive

- Challenges:
  - People want to listen to their music
  - People want to charge their devices
  - Firmware and map data needs to be updated
  - Vehicle systems need to be tested using USB-based tools
  - BadUSB recently highlighted the implicit trust issues around USB


- Solutions:
  - Ensure all firmware/map updates have been cryptographically signed and the software that performs the code signing checks has been rigorously tested
  - Ensure devices drivers for vendor testing tools are removed from production vehicles
  - Ensure all media parsing code has been rigorously security tested
  - Ensure the USB driver stack has been rigorously security tested
  - Ensure only required USB class drivers are installed

Image: leadershipfreak.files.wordpress.com

# Security through obscurity – Lessons from the IT world

- Time and time again we see this in embedded systems
- Hidden “Security mechanisms” designed to authenticate users who are allowed to update firmware
- Hidden USB class drivers that are used for vendor tools
- Hidden engineering menus that assist with fuzz testing instrumentation
- Just because something is hidden doesn’t mean it is protected!


Image: http://farm8.static.flickr.com

# Conclusions

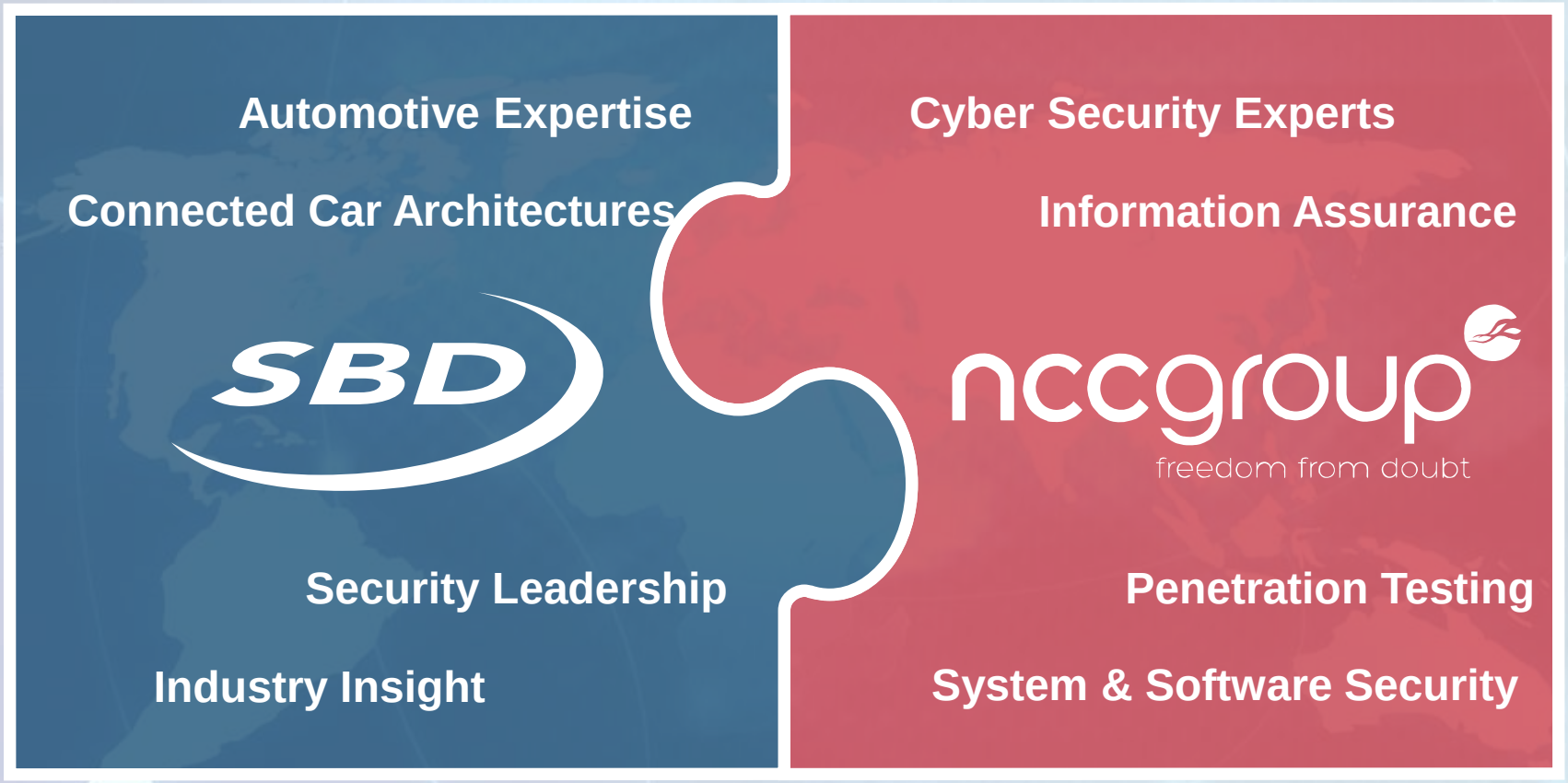
- The USB protocol is already in many vehicles and is here to stay
- USB within vehicles may not have been previously considered a serious attack vector
- I have shown that the impact of successful USB attack can be serious
- Just because physical access is required doesn’t mean the threat can be ignored
- As users continue to want new functionality and creative ways to interact with their vehicles, the attack surface will continue to increase

# Questions?

**Andy Davis, Research Director NCC Group**

**andy.davis ‘at’ nccgroup ‘dot’ com**

# The Unique Automotive Cyber Security Partnership




info@sbd.co.uk

info@nccgroup.com

## An engineering approach to cyber security for Automotive