MWR INFOSECURITY

USB Attacks: Fun with Plug and 0wn

T2'09

Rafael Dominguez Vega

29th October 2009

A little about me ...

rafael.dominguez-vega[at]mwrinfosecurity[dot]com

http://labs.mwrinfosecurity.com

Main Objectives

- Attacks & Impact
- Bug Discovery
- Driver Exploitation

What this talk will cover

- USB for fun and profit
- Fuzzing Techniques
- Crash Debugging
- Exploitation
- Hardware Implementation
- A few demos here and there....

MWR LABS

Once upon a time ...

USB Attacks

- AutoRun (Conficker...)
- Mislaid or Planted Devices
- Driver Bugs

USB Attacks (cont.)

- AutoRun Disabled
- Encrypted USB Pen Drives
- USB Bus Disabled

How Pwnies at home became ‘Research’...

- There was a ‘problem’ target (a client)
- Hardware/Software Testing
- New Feature – USB port implemented

USB Driver Testing

- Black Box Testing

- White Box Testing

MWR LABS

And of course... Beer Based Testing!

USB Technical Background

- USB Communication
  - Enumeration
  - Descriptors
  - Other ‘protocols’

Enumeration

- Device Identification
- Automatic
- Descriptors Sent

Descriptors

- Device Descriptor
- Configuration Descriptor
- Interface Descriptor
- Endpoint Descriptor
- String Descriptor

Device Descriptor

```
const USB_DEVICE_DESCRIPTOR DeviceDescriptor = {
    sizeof(USB_DEVICE_DESCRIPTOR), /* bLength */
    TYPE_DEVICE_DESCRIPTOR,        /* bDescriptorType */
    0x0110,                        /* bcdUSB USB Version 1.1 */
    0,                             /* bDeviceClass */
    0,                             /* bDeviceSubclass */
    0,                             /* bDeviceProtocol */
    8,                             /* bMaxPacketSize 8 Bytes */
    0xBEEF,                        /* idVendor */
    0x1337,                        /* idProduct */
    0x0000,                        /* bcdDevice */
    1,                             /* iManufacturer String Index */
    0,                             /* iProduct String Index */
    0,                             /* iSerialNumber String Index */
    1                              /* bNumberConfigurations */
};
```

Refer. Microchip Technology Inc. Low Pin Count USB Development Kit User's Guide

String Descriptor

```
//Manufacturer string descriptor
ROM struct{BYTE bLength;BYTE bDscType;WORD string[12];}
sd002={sizeof(sd002),USB_DESCRIPTOR_STRING,
{
'M','A','N','U','F','A','C','T','U','R','E','R'
}};

//Product string descriptor
ROM struct{BYTE bLength;BYTE bDscType;WORD string[7];}
sd003={sizeof(sd003),USB_DESCRIPTOR_STRING,
{
'P','R','O','D','U','C','T'
}};
```

Refer. Microchip Technology Inc. Low Pin Count USB Development Kit User's Guide

USB Driver Fuzzing

- ‘Real’ hardware (Expensive)
- Virtualised (QEMU)
- USB over IP (WCPGW)
- Hardware Fuzzer (It’s cool :-P)

QEMU Testing

- Open Source
- Machine Emulator & Virtualiser
- USB Emulation

MWR LABS

QEMU Testing (cont.)

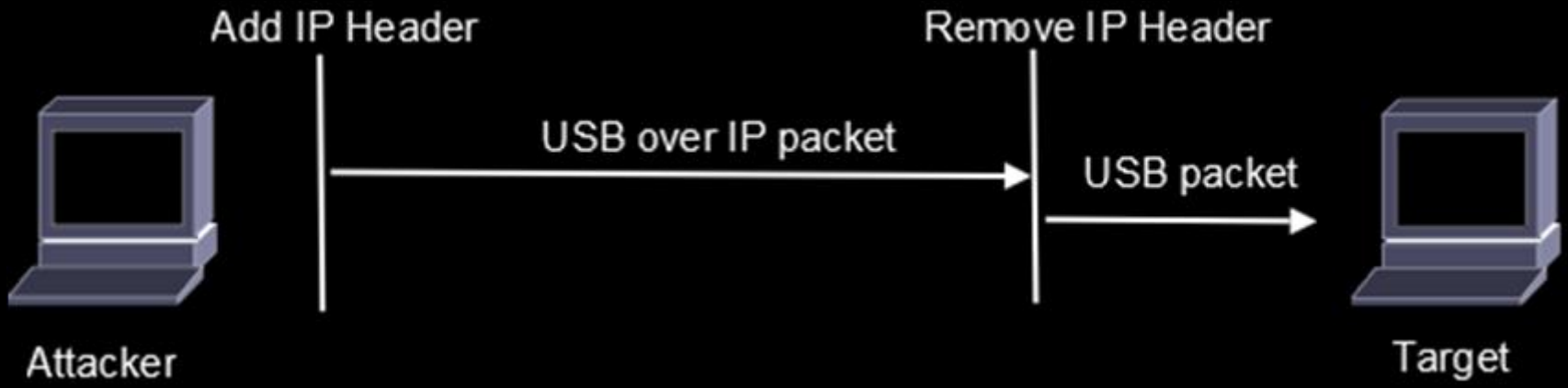
QEMU Testing (cont. II)

- Advantages
  - Quick Start Up
  - Low Resources
  - ‘Oops’ doesn’t trash hardware.
- Disadvantages
  - Fuzzing Engine
  - Re-compile

USB over IP Fuzzing

- USB/IP
- Encapsulate USB packets
- IP Headers

USB over IP Fuzzing (cont.)

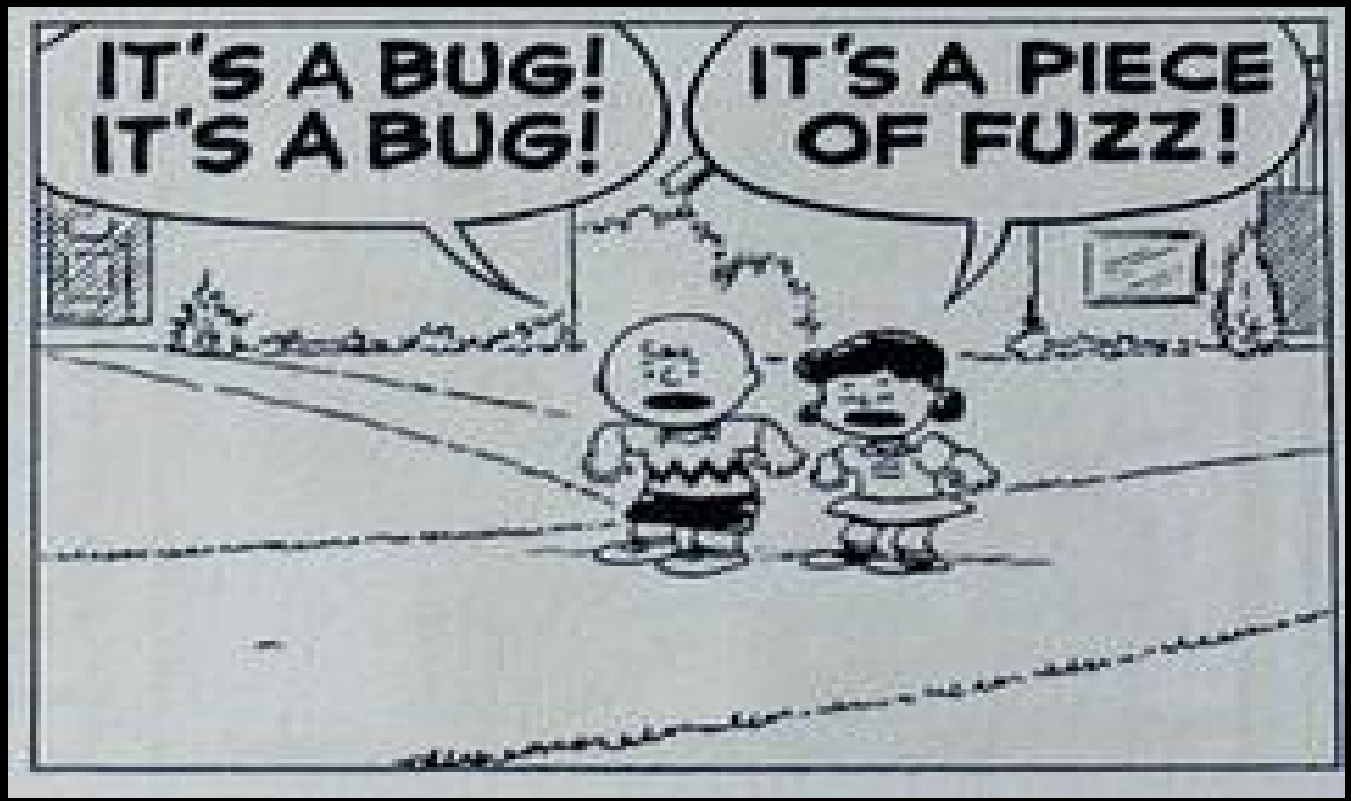


USB over IP Fuzzing (cont. II)

- Advantages
  - Fuzzing Engine
- Disadvantages
  - Reliance on the software

Hardware Fuzzer

- More Reliable
- Much cooler!
- Directly Fuzzing using Hardware
- Man-in-the-middle
- Longer Term Project


IT'S A BUG!
IT'S A BUG!
IT'S A PIECE OF FUZZ!

MWR LABS

Linux USB Driver Bug

Linux USB Driver Bug (cont.)

- auerswald.c
- auerswald_probe function
- Buffer Overflow
- Enumeration Phase
- String Descriptor

Linux USB Driver Bug (cont. II)

```
/* Try to get a suitable textual description of the device */
/* Device name:*/
ret = usb_string( cp->usbdev, AUSI_DEVICE, cp->dev_desc, AUSI_DLEN-1);
 if (ret >= 0) {
  u += ret;
  /* Append Serial Number */
  memcpy(&cp->dev_desc[u], ",Ser# ", 6);
  u += 6;
  ret = usb_string( cp->usbdev, AUSI_SERIALNR, &cp->dev_desc[u], AUSI_DLEN-u-1)
  if (ret >= 0) {
        u += ret;
        /* Append subscriber number */
        memcpy(&cp->dev_desc[u], ", ", 2);
        u += 2;
        ret = usb_string( cp->usbdev, AUSI_MSN, &cp->dev_desc[u], AUSI_DLEN-u-1
        if (ret >= 0) {
                u += ret;
        }
  }
```

Linux USB Driver Bug (cont. III)

- Element of Device Structure
- usb_string function
- Overwrite other elements of structure

MWR LABS

Kernel Crash Demo

Crash Analysis

- GDB
- Crash Utility
- KGDB

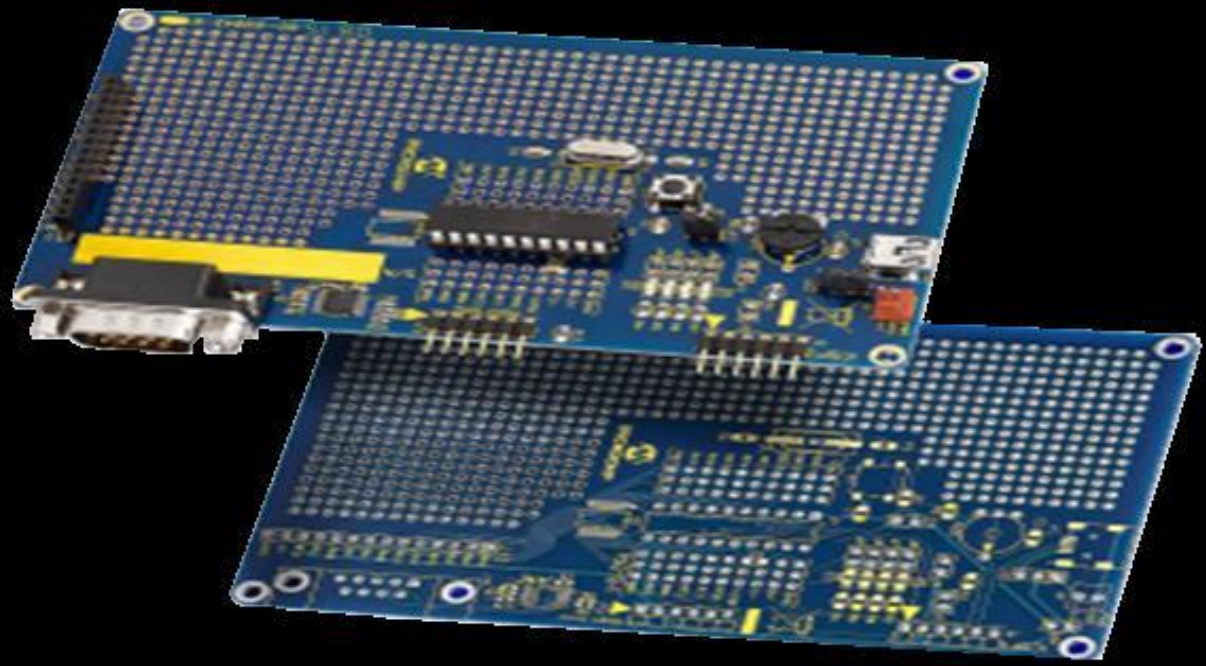
KGDB (cont.)

Hardware Implementation

- PIC18 Family Microcontroller
- Malicious Auerswald Device
- Flash Microcontroller with Shellcode
- Exploit Driver Bug

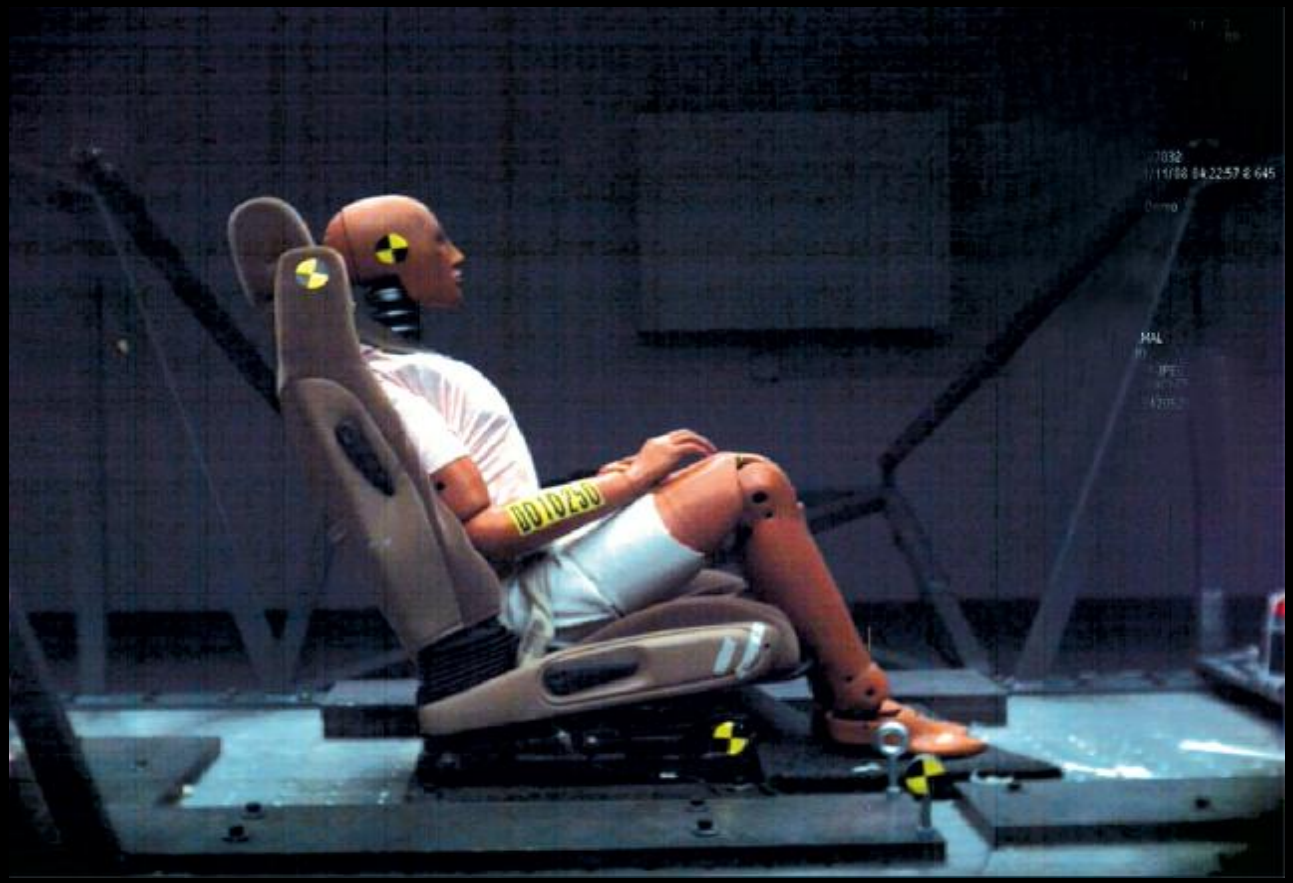
MWR LABS

PIC18F14K50



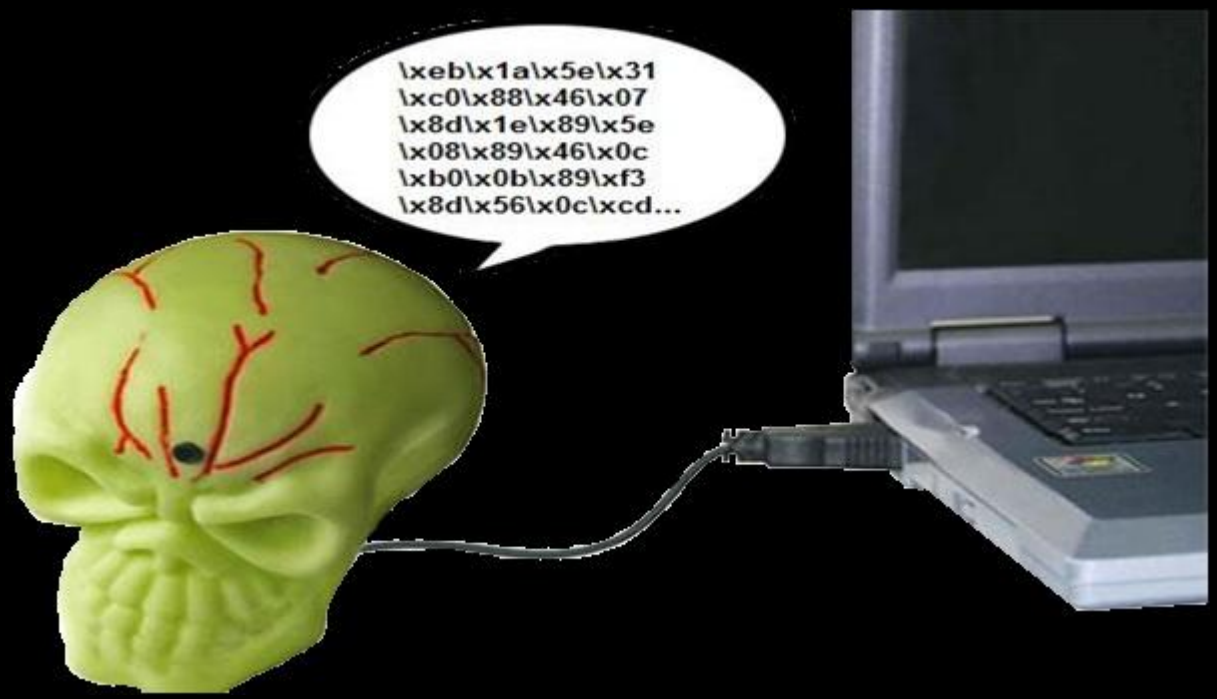
MWR LABS

Crash Analysis & Exploit - Demo



MWR LABS

Exploiting Driver Bug with Malicious USB Device (Demo)



Recommendations

- Disable not required USB drivers
- Security Test USB Drivers
- Assess USB Risks

References & Further Reading

USB Official Site
http://www.usb.org/

Linux USB
http://www.linux-usb.org/

Microchip Technology Inc.
http://www.microchip.com/

Microchip Technology Inc.

- Low Pin Count USB Development Kit User's Guide
- PIC18F13K50/14K50 Data Sheet

Beyond Logic
http://www.beyondlogic.org/

USB Design by Example: A Practical Guide to Building I/O Devices (Intel University Press) by John Hyde

References & Further Reading (cont.)

QEMU
http://www.qemu.org/

USB/IP
http://usbip.sourceforge.net/

White Paper: Red Hat Crash Utility
http://people.redhat.com/anderson/crash_whitepaper/

KGDB: Linux Kernel Source Level Debugger
http://kgdb.linsyssoft.com/

Evaluating Security Aspects of the Universal Serial Bus
http://www.informatik.uni-hamburg.de/SVS/archiv/slides/09-01-13-OS-Jodeit-Evaluating_Security_Aspects_of_USB.pdf

MWR LABS

I'll get by with a little help from my friends...

http://labs.mwrinfosecurity.com