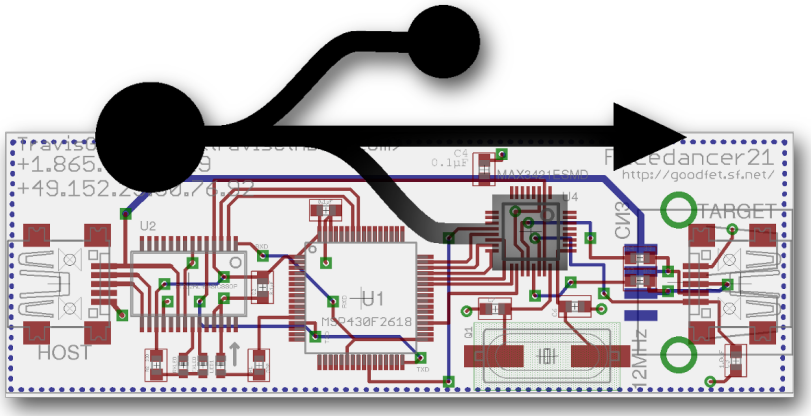
USB-Sicherheit

Finden und Ausnutzen von Schwachstellen in USB-Host-Implementierungen

Gerhard Klostermeier

Hochschule Aalen, in Kooperation mit SySS GmbH

17. Oktober 2014



Inhalt

1. USB
2. Schwachstellen Finden und Analysieren
3. Angriffe
4. Weitere Funde
5. Fazit

Inhalt

1. USB
   - Allgemein
   - USB-Standards
   - USB-Modell
   - Enumeration & Descriptors
2. Schwachstellen Finden und Analysieren
3. Angriffe
4. Weitere Funde
5. Fazit

Allgemein
USB

- *Universal Serial Bus*
- Ursprünglich von Intel entwickelt
- Markteinführung: 1996
- Ziel: Schnittstelle für verschiedene Peripheriegeräte

USB-Standards I

USB 1.0

- Januar 1996
- Geschwindigkeit: 1.5 Mbit/s (*Low Speed*), 12 Mbit/s (*Full Speed*)
- Nur wenige Geräte schafften es auf den Markt

USB 1.1

- September 1998
- Farbe: Weiß
- Behebt Probleme und Unklarheiten von USB 1.0
- Geräteanzahl auf dem Markt nimmt zu, aber:
  - Übertragungsgeschwindindigkeit unzureichend (z.B. für Video)
  - Mangelhafte oder fehlende USB-Stack-Implementierung

USB-Standards II

USB 2.0

- April 2000
- Geschwindigkeit: 12 Mbit/s (*Full Speed*), 480 Mbit/s (*Hi-Speed*)
- Farbe: Schwarz
- Spätere Erweiterungen (Auszug):
  - 2001, Dez.: Mini-B Steckverbinder
  - 2001: On-The-Go
  - 2007: Micro-USB Steckverbinder 1.01

USB 3.0

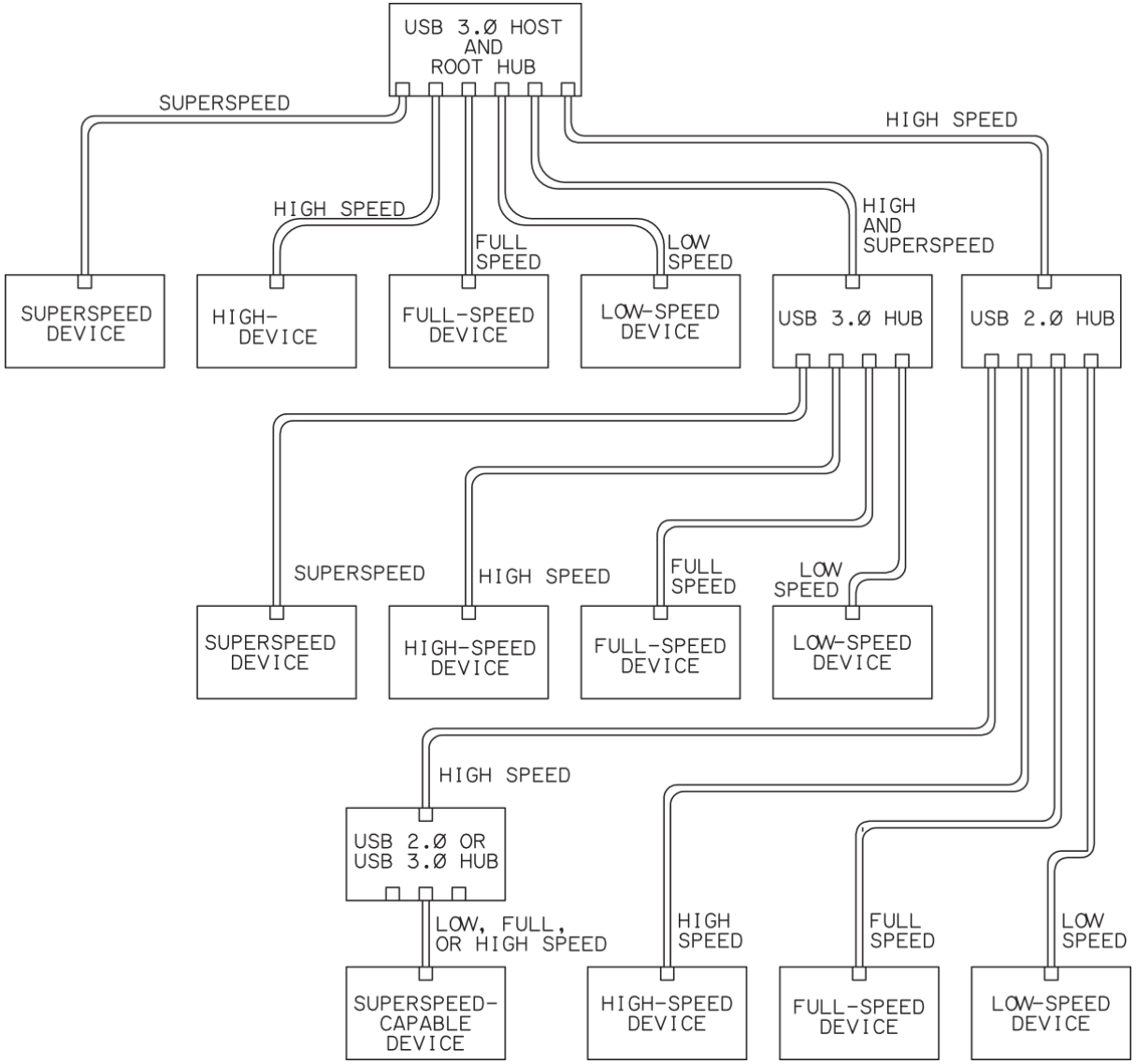
- November 2008
- Geschwindigkeit: 5 Gbit/s (*SuperSpeed*)
- Farbe: Blau

USB-Modell I

- Host / Peripherie
- Host ist „Master“ (*Host-Based*)
- Sterntopologie (mit *branching* durch Hubs)
- Übertragungsart: Seriell, mit Protokoll
- Maximal 127 Clients pro Hostcontroller
- Maximal 5 mal kaskadiert

USB-Modell II

USB



1

[1] Quelle: USB Complete, Jan Axelson, 2009, ISBN: 978-1931448086

Enumeration

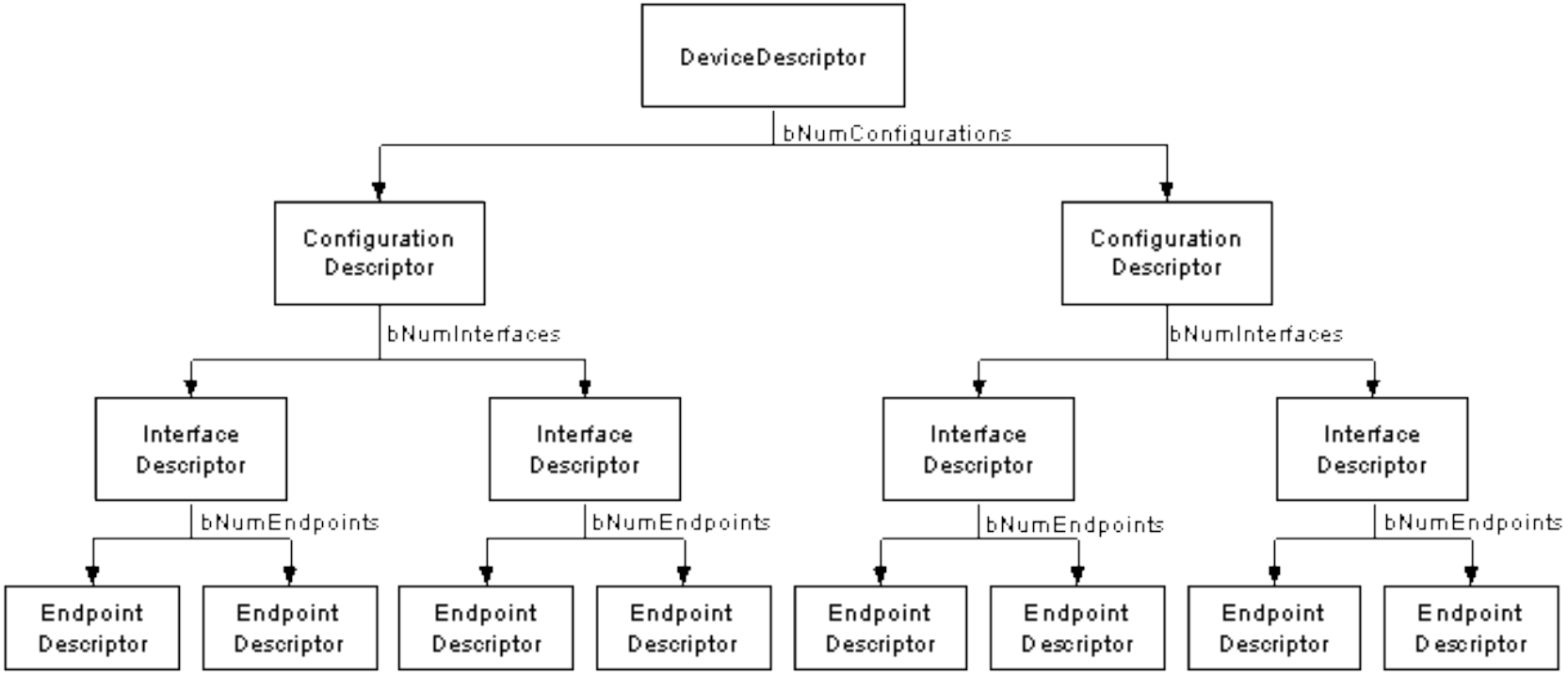
- USB-Geräte erkennen und konfigurieren
- Kommunikation über Default Pipe, EP0
- Maximaldauer der Phase: 5 Sekunden
- Standard Requests:
  - Standard Device Request
  - Standard Interface Requests
  - Standard Endpoint Requests

Deskriptoren

- Nach USB-Standard formatierte Informationseinheit
- Beschreiben ein USB-Gerät
- Werden in der Enumerationsphase abgefragt
- Hierarchisch aufgebaut

Hierachischer Aufbau

Enumeration & Descriptors



[2]

[2] Quelle: http://www.beyondlogic.org/usbnutshell

Device Descriptor I

- Beschreibt das USB-Gerät als Ganzes
- Es gibt nur einen Device Descriptor

Device Descriptor II

Offset (Dezimal)	**Feld**	**Größe** (Byte)	**Beschreibung**
0	`bLength`	1	Größe des Deskriptors in Byte
1	`bDescriptorType`	1	Device Descriptor: 01_h
2	`bcdUSB`	2	Release-Nummer der USB-Spezifikation (BCD, z.B. 0110_h = USB 1.1)
4	`bDeviceClass`	1	Klassencode
5	`bDeviceSubClass`	1	Unterklassencode
6	`bDeviceProtocol`	1	Protokollcode
7	`bMaxPacketSize(0)`	1	Maximale Paketgröße für den Endpunk 0
8	`idVendor`	2	Anbieterkennung
10	`idProduct`	2	Produktkennung
12	`bcdDevice`	2	Release-Nummer des Geräts (BCD)
14	`iManufacturer`	1	Index des String Descriptors für den Hersteller
15	`iProduct`	1	Index des String Descriptors für das Produkt
16	`iSerialNumber`	1	Index des String Descriptors für die Seriennummer
17	`bNumConfigurations`	1	Anzahl möglicher Konfigurationen

Configuration Descriptor I

- Beschreibt die elektrischen Eigenschaften eines USB-Geräts
- Es kann mehrere Configuration Deskriptoren geben
- In Praxis gibt es meistens nur eine Konfiguration

Configuration Descriptor II

Offset (Dezimal)	**Feld**	**Größe** (Byte)	**Beschreibung**
0	`bLength`	1	Größe des Deskriptors in Byte
1	`bDescriptorType`	1	Configuration Descriptor: 02_h
2	`wTotalLength`	2	Größe des Configuration Descriptors und aller untergeordneten Deskriptoren in Byte
4	`bNumInterfaces`	1	Anzahl der Interfaces dieser Konfiguration
5	`bConfigurationValue`	1	Identifizierer für Standard Requests (`Set_Configuration` und `Get_Configuration`)
6	`iConfiguration`	1	Index des String Descriptors für die Konfiguration
7	`bmAttributes`	1	Self-Powered/Bus-Powered und Remote-Wakeup-Einstellungen
8	`bMaxPower`	1	Benötigter Busstrom in 2 mA (USB2.0) oder 8 mA (SuperSpeed) Schritten

Interface Descriptor I

- Beschreibt eine Funktion des USB-Geräts
- Ein Gerät kann mehrere Funktionen (Interface Descriptor) haben (z.B. Webcam → Video und Audio)
- Funktionen können alternative Einstellungen haben (z.B. Audio → Mono oder Stereo)

Interface Descriptor II

Offset (Dezimal)	**Feld**	**Größe** (Byte)	**Beschreibung**
0	`bLength`	1	Größe des Deskriptors in Byte
1	`bDescriptorType`	1	Interface Descriptor: 04_h
2	`bInterfaceNumber`	1	Nummer zu Identifizierung
3	`bAlternateSetting`	1	Ein zur Auswahl einer alternativen Einstellung verwendeter Wert
4	`bNumEndpoints`	1	Anzahl der Endpunkte (Endpoint 0 wird nicht mitgezählt)
5	`bInterfaceClass`	1	Klassencode
6	`bInterfaceSubClass`	1	Unterklassencode
7	`bInterfaceProtocol`	1	Protokollcode
8	`iInterface`	1	Index des String Descriptors für das Interface

Endpoint Descriptor I

- Beschreibt die Eigenschaften des Kommunikationsendpunkts
- Legt die Kommunikationsrichtung und -art fest
- Es kann 0, 1 oder mehere Endpoints geben
  (max. 3 für USB 1.x, max. 16 für USB 2.0+)
- Endpoint 0 gibt es auch ohne Deskriptor immer
  (wird von der USB-Spezifikation vorausgesetzt)

Endpoint Descriptor II

Offset (Dezimal)	**Feld**	**Größe** (Byte)	**Beschreibung**
0	`bLength`	1	Größe des Deskriptors in Byte
1	`bDescriptorType`	1	Endpoint Descriptor: 05_h
2	`bEndpointAddress`	1	Endpunktnummer und -richtung
3	`bmAttributes`	1	Unterstützter Transfertyp
4	`wMaxPacketSize`	2	Maximal unterstützte Paketgröße
6	`bInterval`	1	Maximale Latenz/Abfrageintervall/NAK-Rate

String Descriptor

- Beschreiben Device, Configurations und Interfaces durch lesbare Zeichenketten
- Optional und hierarchisch unabhängig von den anderen Deskriptoren
- Beispiel: „Logitech, Inc. G3 (MX518) Optical Mouse“

Class Codes I

- In Abhänigkeit der Class Codes gibt es weitere Deskriptoren
- Klasse wird durch `bDeviceClass`, `bDeviceSubClass` und `bDeviceProtocol` festgelegt
- Falls `bDeviceClass` = 0, `bDeviceSubClass` = 0 und `bDeviceProtocol` = 0 wird die Klasse in `bInterfaceClass`, `bInterfaceSubClass` und `bInterfaceProtocol` festgelegt

Class Codes II

Enumeration & Descriptors

Klassencode (Hexadezimal)	**Beschreibung**
00	Reserviert
01	Audio
02	Communication device class: Communictaion interface (CDC Control)
03	Human Interface Device (HID)
05	Physical
06	Image
07	Printer
08	Mass storage
09	Hub
0A	Communicaiton device class: Data interface (CDC Data)
0B	Smart Card
0D	Content Security
0E	Video
0F	Personal healthcare device
10	Video/Audio device
DC	Diagnostic device
E0	Wireless controller
EF	Miscellaneous
FE	Application specific
FF	Vendor specific

Inhalt

1. USB
2. Schwachstellen Finden und Analysieren
   - Methoden
   - Hardware
   - Software
3. Angriffe
4. Weitere Funde
5. Fazit

Methoden

Schwachstellen Finden und Analysieren

- Black Box Testing
- Fuzzing
- Code Review
- Static Code Analysis
- Reverse Engineering
- Debugging
- Monitoring

Facedancer I

Hardware - Schwachstellen Finden und Analysieren

3

Facedancer II

Name

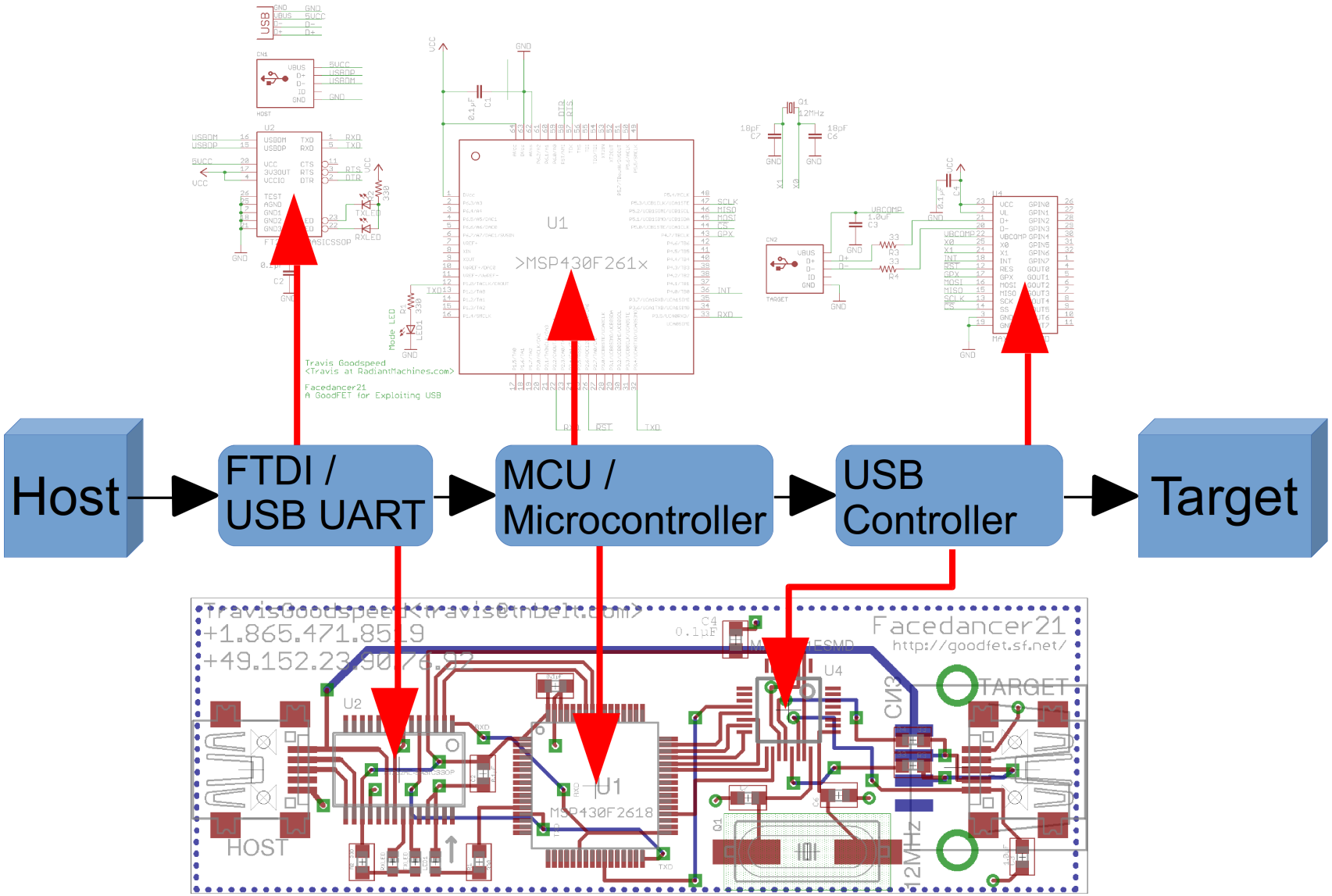
In Frank Herberts *Dune* Universum sind „Face Dancer“ Formwandler welche nach Belieben ihr Äußeres ändern können.

Allgemein:

- Board um bequem (z.B. mit Python) USB-Geräte zu emulieren
- Erleichtert Entwicklung neuer USB-Geräte
- Kann zum Testen von USB-Implementierungen genutzt werden
- Entwickelt von Travis Goodspeed (und Community)

Facedancer III

Hardware - Schwachstellen Finden und Analysieren



Facedancer IV

FTDI

- FT232RL
- Von USB auf serielle Schnittstelle (RS-232) umwandeln

Microcontroller

- MSP430F2618
- Serielle „Befehle“ für den USB Controller „übersetzen“ (MAXUSB)
- Weitere Funktionen der Firmware: Monitor und SPI

USB Controller

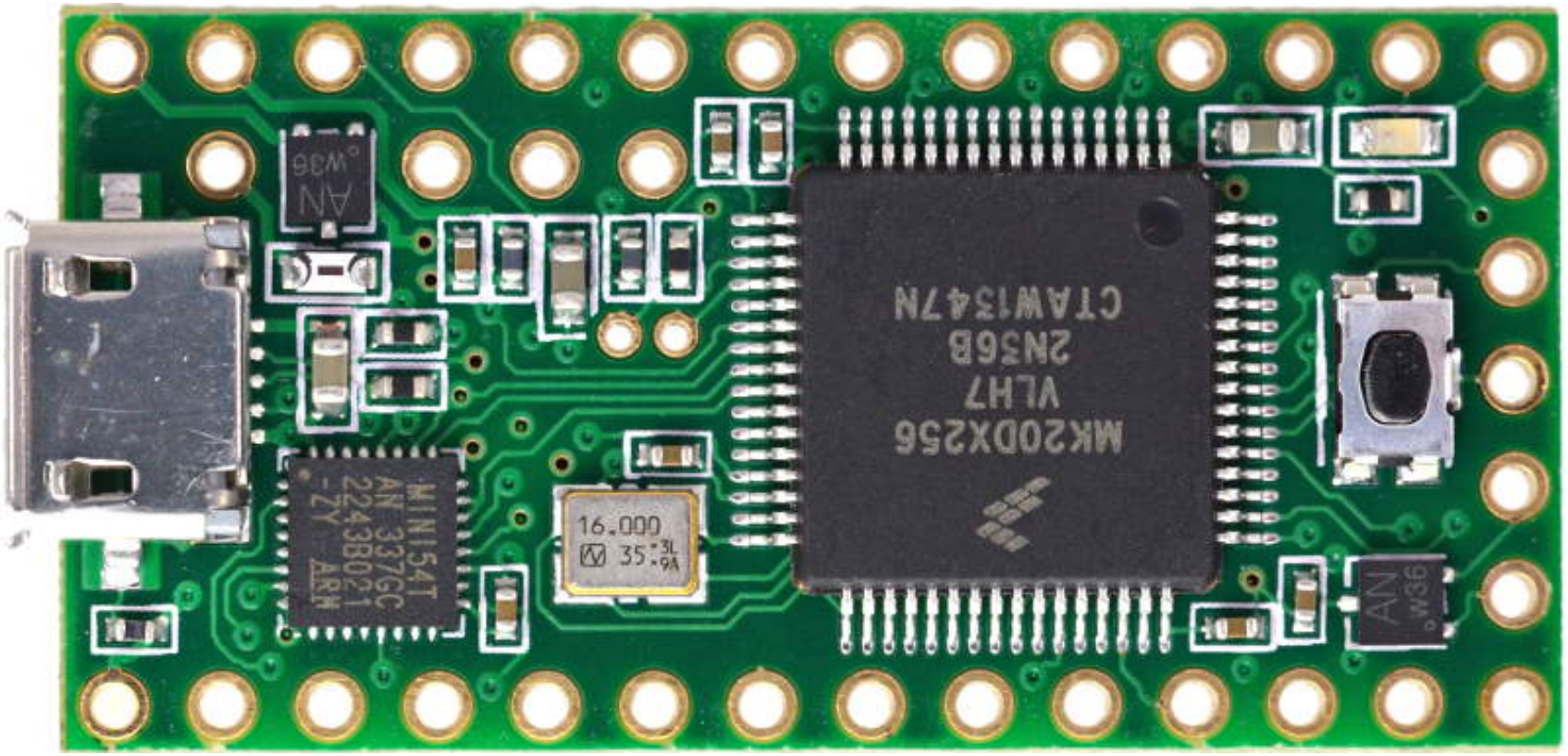
- MAX3421E
- Unterstützt Host und Client Modus (USB 2.0)
- USB connect/disconnect programmierbar
- Über SPI ansteuerbar

[3] Quelle: http://int3.cc/products/facedancer21

Teensy Board

Hardware - Schwachstellen Finden und Analysieren

„The Teensy USB Development Board is a complete USB-based microcontoller development system.“ – pjrc.com

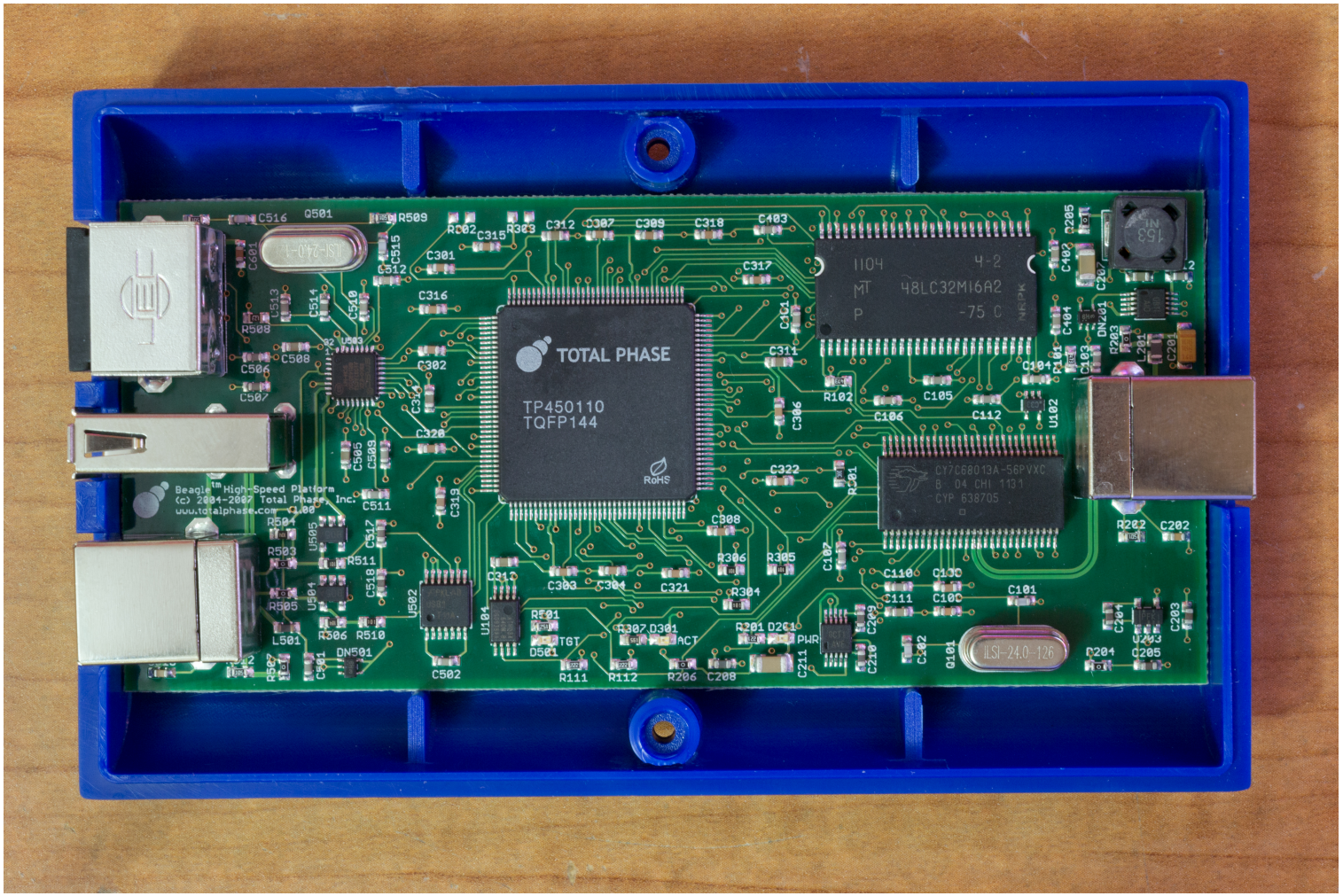


[4]

[4] Quelle: https://www.pjrc.com/store/teensy31.jpg

Beagle USB 480 Protocol Analyzer

Hardware - Schwachstellen Finden und Analysieren



[5]

[5] Quelle: https://secure.flickr.com/photos/eric_agan/7140329925/

Software

- umap
  - Fuzzer für den Facedancer
  - Entwickelt von Andy Davis (NCC Group)
  - https://github.com/nccgroup/umap
- IDA Pro
  - *Interactive Disassembler* (IDA) für viele Prozessorarchitekturen
  - http://www.hex-rays.com/idapro
- WinDbg – Debugger & Crash-Analyse-Werkzeug für Windows
- Network Monitor – Netzwerkanalysewerkzeug für Windows
- Wireshark – Netzwerkanalysewerkzeug (Multi-Plattform)
- DriverView – Infos über geladene Treiber (für Windows)
- gdb – Debugger für Linux
- crash – Linux Crash Dumps analysieren
- lsusb – Infos über angeschlossene USB-Geräte anzeigen (Linux)
- ...

Inhalt

1. USB
2. Schwachstellen Finden und Analysieren
3. Angriffe
   - Linux DoS
   - Windows DoS
   - Windows Domain DoS
   - Teensy als Tastatur
   - Windows RNDIS-Treiber (MS13-027)
   - Linux Audiotreiber (CVE-2011-0712)
4. Weitere Funde
5. Fazit

Linux DoS

Angriffe

- X11-Crash durch Formatstring-Angriff
- Ubuntu 12.04 (und Andere)
- Im Manufacturer oder Product String Descriptor
- Beispiel: „`Keyboard %n%n%n%n`“

Windows DoS

Angriffe

- Windows Crash (BlueScreen) beim Parsen von HID Deskriptoren
- Windows XP, Vista, 7, 8, 8.1 (32 und 64 Bit)
- Null-Pointer Dereference
- Im Report Descriptor eines HID
- Beispiel: `USAGE_PAGE` einer USB-Tastatur von `0x0501` auf `0x8101` setzen

Demo

Angriffe

Demo.

Windows Domain DoS I

Angriffe

- Windows Crash (BlueScreen)
- Windows XP, Vista, 7, 8, 8.1, Server 2012
- Im USB-Audio-Treiber (*usbaudio.sys*)

- USB Redirection via RDP:
  USB-Daten werden abstrahiert an Server weitergeleitet
  („High-Level-Protokoll“)
- RemoteFX USB Redirection:
  URBs werden direkt an Server weitergeleitet

USB Redirection via RDP

RemoteFX USB Redirection

[6]

[6] Quelle: https://www.nccgroup.com/media/481264/usb_physical_access_whitepaper_v1-0.pdf

Teensy als Tastatur

- Teensy gibt sich als Tastatur aus
- Beliebige Tastenanschläge können getätigt werden
- Teensy Board wird vom Angreifer in ein unauffälliges Gehäuse gesetzt (z.B. USB-Stick)
- Vorteil: Fast jedes System lässt sich vollständig mit Tastatur (und Maus) bedienen
- Nachteil: Opfer kann die Eingaben am Bildschirm verfolgen

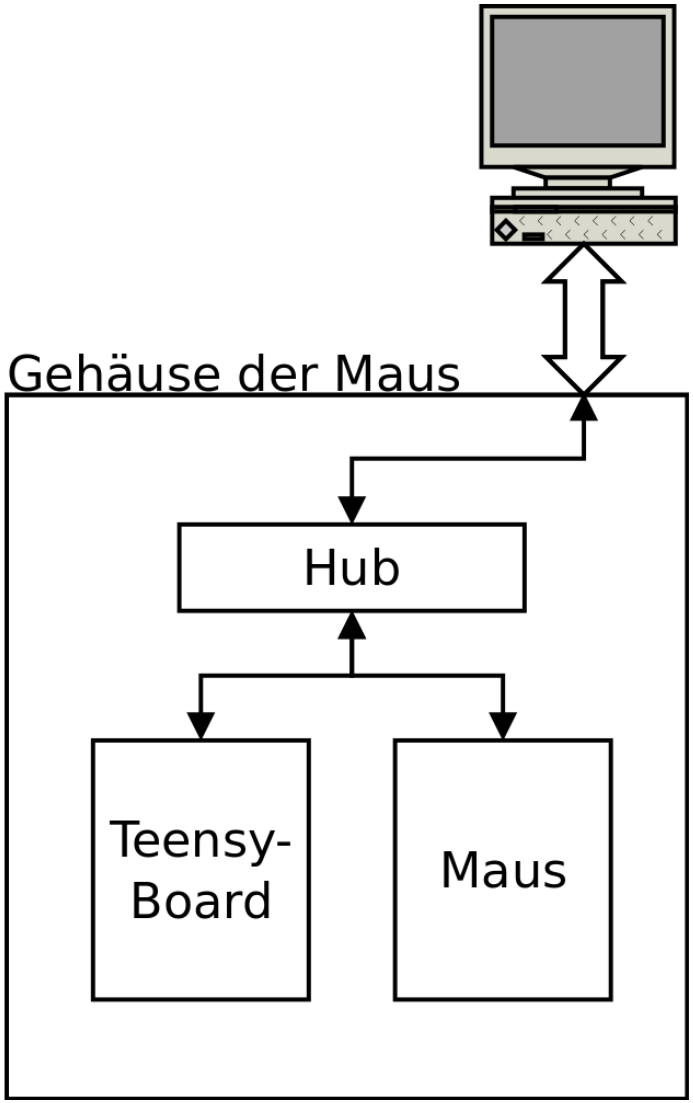
Demo

Angriffe

Demo.

Trojanische Maus I

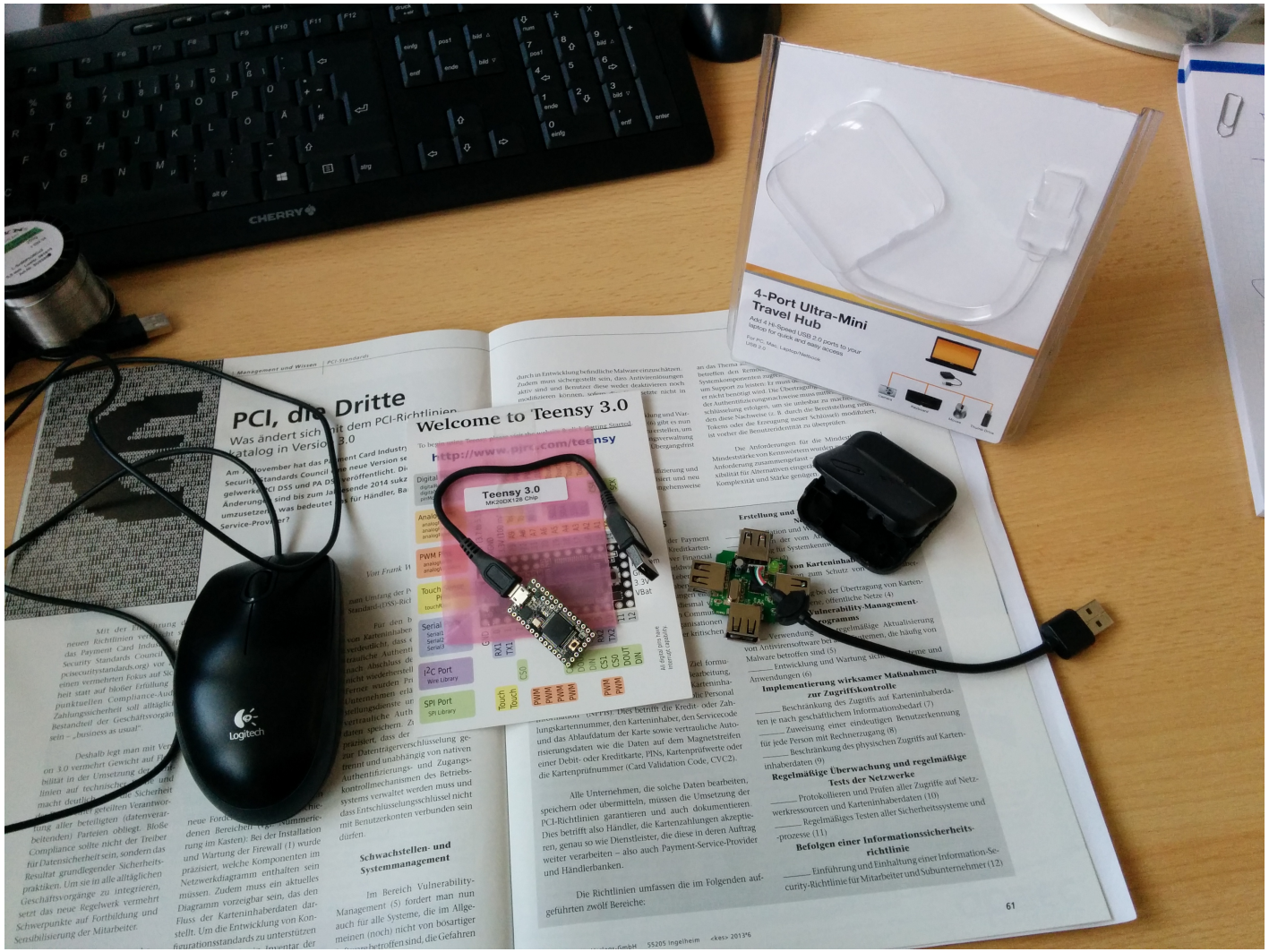
Angriffe



7

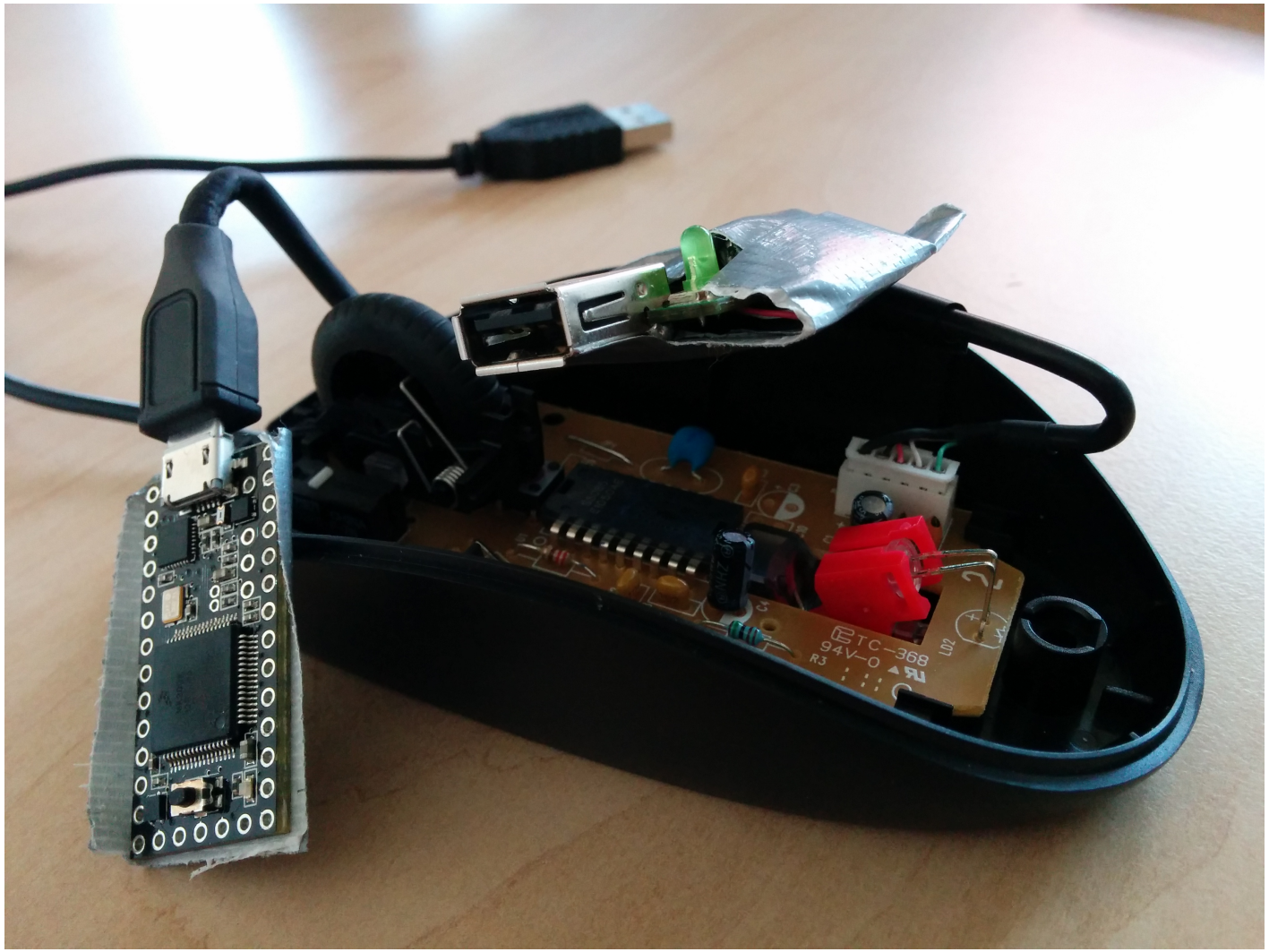
Trojanische Maus II

Angriffe



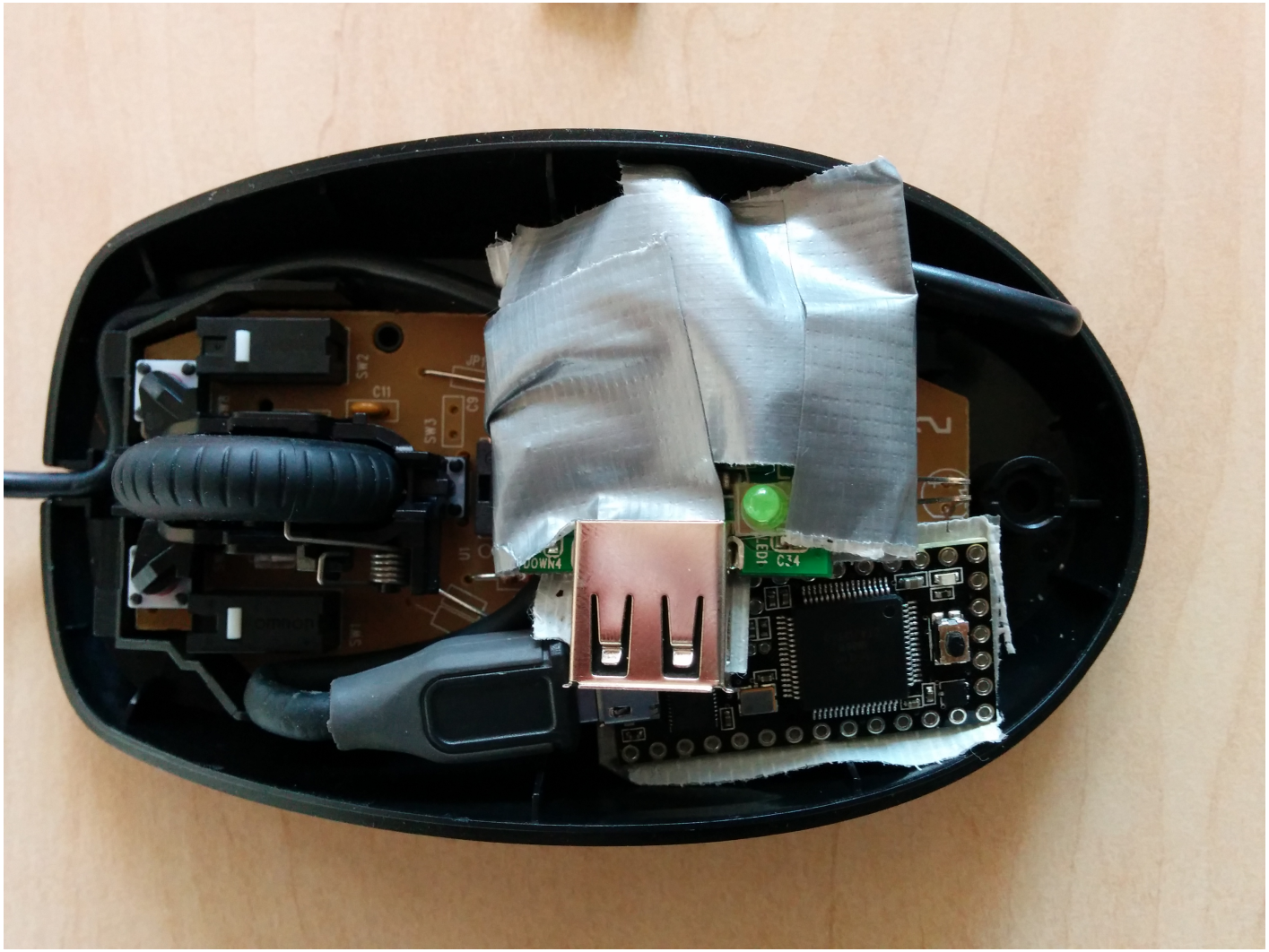
Trojanische Maus III

Angriffe



Trojanische Maus IV

Angriffe



[7] Quelle: http://www.trojanmice.com/images/TROJAN2.JPG

Windows RNDIS-Treiber (MS13-027) I

- „Die Sicherheitsanfälligkeit wird dadurch verursacht, dass der Windows-Kernelmodustreiber Objekte im Speicher falsch verarbeitet.“
- „Diese Sicherheitsanfälligkeit kann für eine lokale Erhöhung von Berechtigungen ausgenutzt werden.“
- Von Andy Davis entdeckt
- Behoben durch Update KB2807986

Windows RNDIS-Treiber (MS13-027) II

Analyse:

- Windows Update separat herunter laden
- Betroffenen Treiber aus Update extrahieren (`usb8023.sys`, `usb8023x.sys`
- Treiber für RNDIS
- `usb8023.sys` ist nur für Legacy Support (hat keine INF)
- `usb8023x.sys` vor und nach Update vergleichen
  → `SelectConfiguration`-Methode wurde verändert
- `SelectConfiguration` mit IDA Pro untersuchen (Reverse Engineering)
- Vereinfacht: In der neuen Version werden die Interface Descriptors „gezählt“ anstatt sich auf die Angabe in `bNumInterfaces` zu verlassen

Windows RNDIS-Treiber (MS13-027) III

Testing & Debugging:

- Mit Facedancer ein RNDIS-Gerät emulieren (Enumeration)
- Windows Crash Dump analysieren
- Windows Remote Kernel Debugging (z.B. über FireWire)
- Ein Byte im Kernel `POOL_HEADER` wird auf 0 gesetzt (z.B. `PreviousSize`, `PoolIndex`, `BlockSize`, `PoolType`, ...)

Problemstelle:

```
        push    0
        push    dword ptr [ebp-4]       Counter (bInterfaceNumber)
        push    edi                     ConfigurationDescriptor
b       call    dword ptr [usb80236!_imp__USBD_ParseConfigurationDescriptor (9b7a702c)]
        test    eax,eax                 Found matching Interface?
        je      usb80236!SelectConfiguration+0xa9 (9b7a50e1)
        mov     al,byte ptr [eax+5]     bInterfaceClass
        mov     ecx,dword ptr [ebp+8]
        mov     byte ptr [ecx+4],al     USBD_INTERFACE_INFORMATION.Class = bInterfaceClass
        jmp     usb80236!SelectConfiguration+0xb0 (9b7a50e8)
        mov     eax,dword ptr [ebp+8]
        mov     byte ptr [eax+4],0      USBD_INTERFACE_INFORMATION.Class = 0
        mov     eax,dword ptr [ebp+8]
        movzx   eax,word ptr [eax]
        add     dword ptr [ebp+8],eax   "USBD_INTERFACE_INFORMATION = USBD_INTERFACE_INFOMATION.Next"
        movzx   eax,byte ptr [edi+4]
        inc     dword ptr [ebp-4]       Increment Counter
        cmp     dword ptr [ebp-4],eax   Counter < bNumInterfaces
        jb      usb80236!SelectConfiguration+0x8e (9b7a50c6)
```

Linux Audiotreiber (CVE-2011-0712) I

- Buffer Overflow in Caiaq-Audiotreiber
- Entdeckt von MWR InfoSecurity
- Statisch allokiertes char-Array wird mit `strcpy()` befüllt
- `strcpy()`-Quelle ist String Descriptor `iProduct`

Linux Audiotreiber (CVE-2011-0712) II

Angriffe

linux-2.6.38/sound/usb/caiaq/audio.c

```
int snd_usb_caiaq_audio_init(struct snd_usb_caiaqdev *dev)
{
...
ret = snd_pcm_new(dev->chip.card, dev->product_name, 0,
        dev->n_audio_out, dev->n_audio_in, &dev->pcm);
...
dev->pcm->private_data = dev;
strcpy(dev->pcm->name, dev->product_name);
...
}
```

Linux Audiotreiber (CVE-2011-0712) III

Angriffe

linux-2.6.38/include/sound/pcm.h

```
struct snd_pcm {
    struct snd_card *card;
    struct list_head list;
    int device; /* device number */
    unsigned int info_flags;
    unsigned short dev_class;
    unsigned short dev_subclass;
    char id[64];
    char name[80];
    struct snd_pcm_str streams[2];
    struct mutex open_mutex;
    wait_queue_head_t open_wait;
    void *private_data;
    void (*private_free) (struct snd_pcm *pcm);
    struct device *dev; /* actual hw device this belongs to */
#if defined(CONFIG_SND_PCM_OSS) || defined(CONFIG_SND_PCM_OSS_MODULE)
    struct snd_pcm_oss oss;
#endif
};
```

Linux Audiotreiber (CVE-2011-0712) IV

Analyse:

- Mit Teensy Caiaq-Audiogerät emulieren
- Quelltext des Treibers analysieren
- Die ersten Anfragen des Treibers müssen beantwortet werden bis die kritische Stelle im Code erreicht wird
- Einschränkung: Puffer liegt im Heap & String Descriptor können max. 253 Byte enthalten

Inhalt

1. USB
2. Schwachstellen Finden und Analysieren
3. Angriffe
4. Weitere Funde
5. Fazit

Weitere Funde

- VirtualBox USB Plugin (Oracle VM VirtualBox Extension Pack)
- Linux Kernel Panic (im aktuellen Kernel)
- PS3 Crash
- Dell Vostro Laptop BIOS Crash/Freeze
- Drucker Crash
- (XSS in Router-Webinterface)
- ...

Inhalt

1. USB
2. Schwachstellen Finden und Analysieren
3. Angriffe
4. Weitere Funde
5. Fazit

Fazit

[8]

[8] Quelle: http://radiostudent.si/sites/default/files/slike/2014-05-19-offsajd-kdo-bo-%C5%A1tudentu-pla%C4%8Dal-avtobus-27331.jpg

Fragen

Danke. Fragen?

Goodies

HDD aus China I

Goodies

9

HDD aus China II

Goodies

[9]

[9] Quelle: http://www.thaiseoboard.com/index.php?topic=187469.0

WLAN-Stick aus China

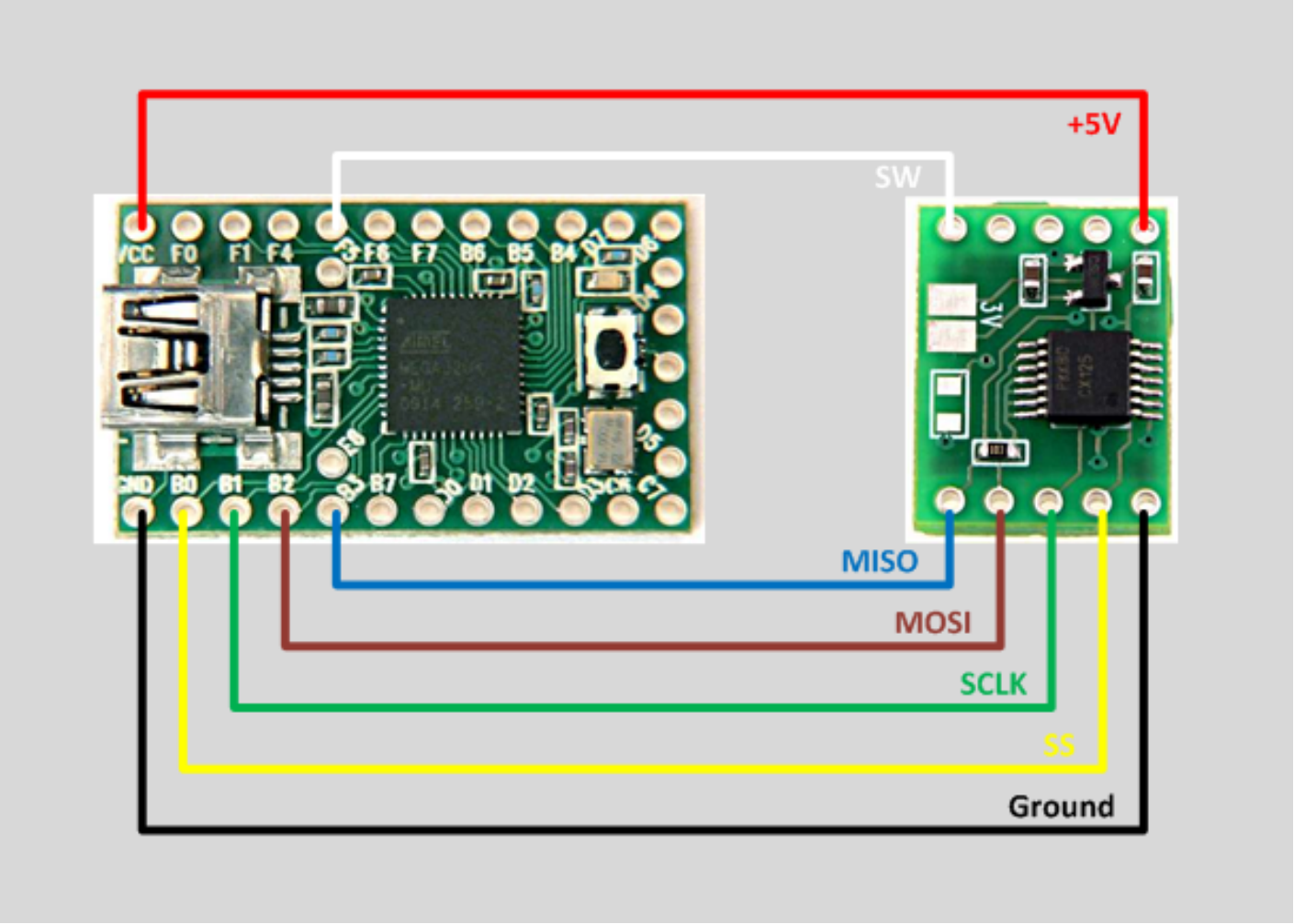
Goodies

[10]

[10] Quelle: http://www.thaiseoboard.com/index.php?topic=187469.0

Flash Drive HW-Trojan I

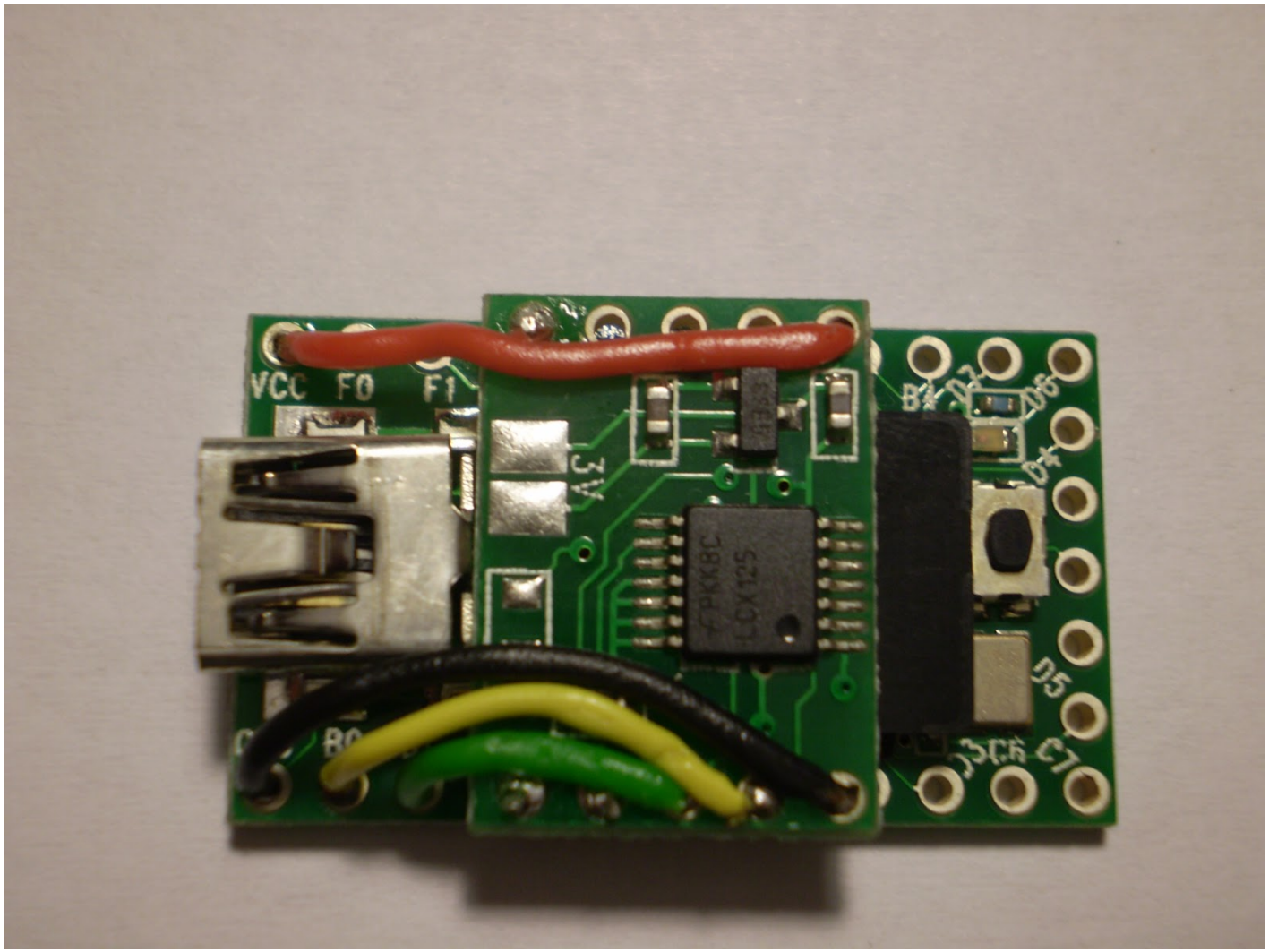
Goodies



11

Flash Drive HW-Trojan II

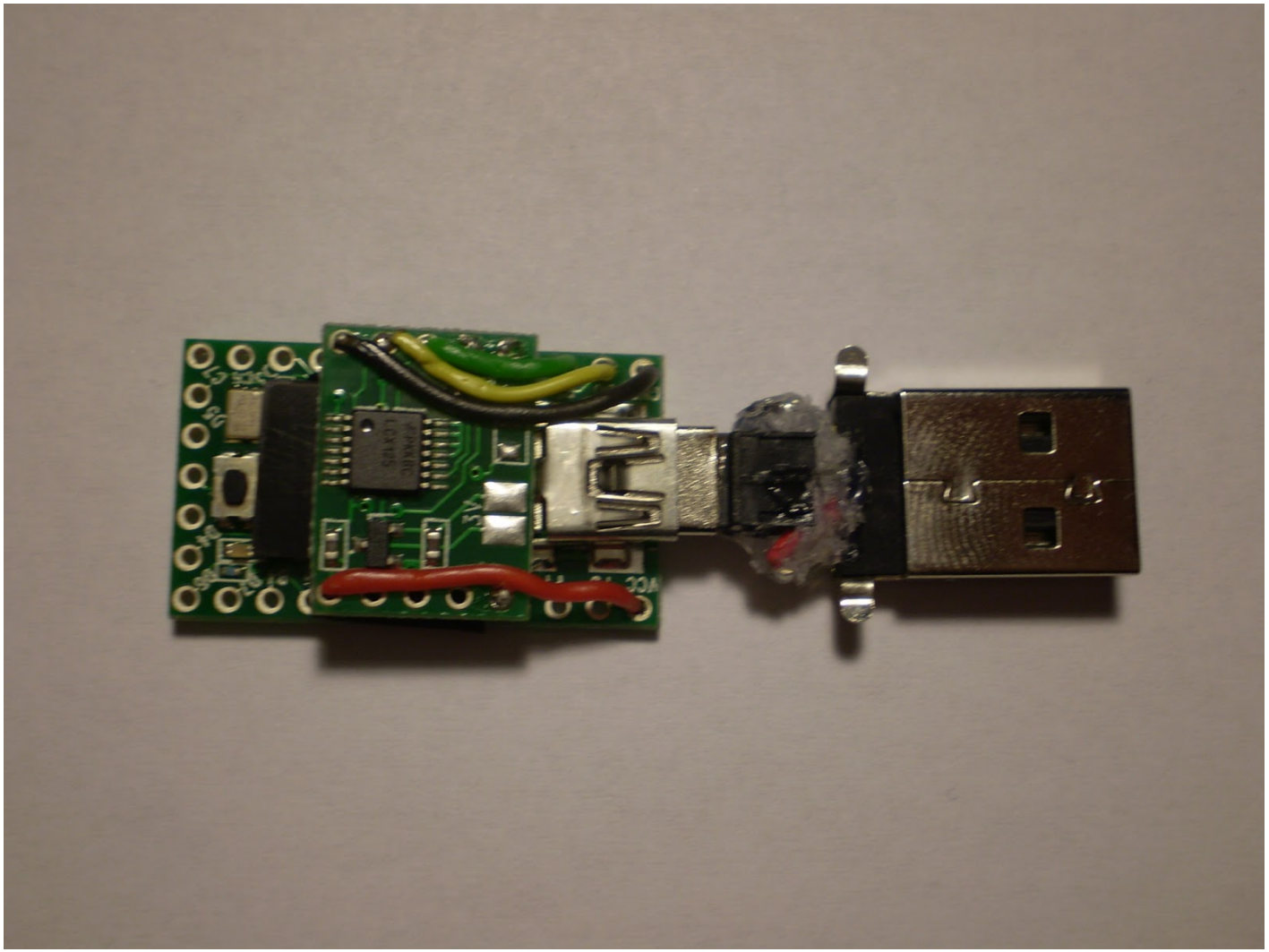
Goodies



11

Flash Drive HW-Trojan III

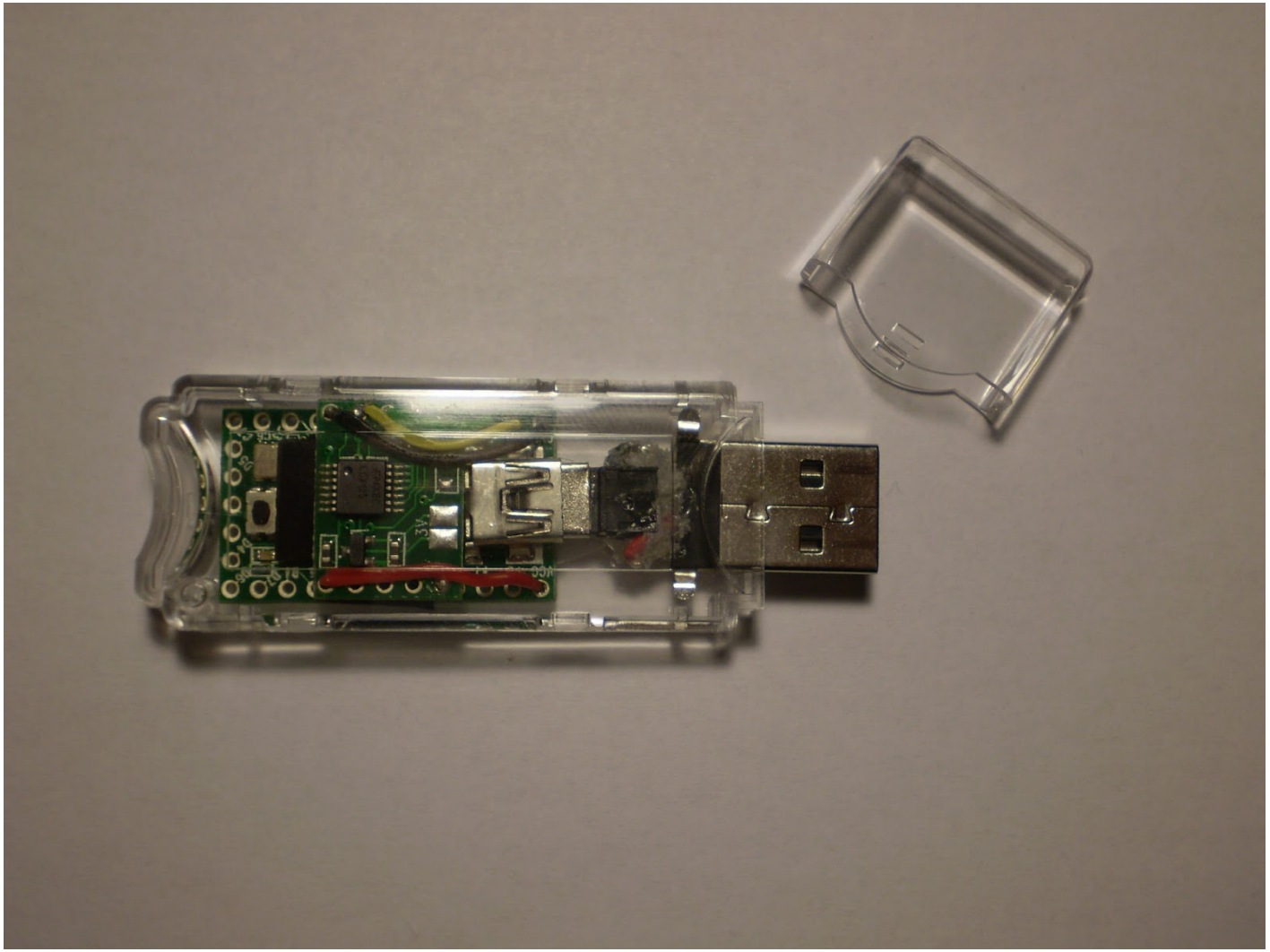
Goodies



11

Flash Drive HW-Trojan IV

Goodies



[11]

[11] Quelle: http://jumpespjump.blogspot.be/2013/01/making-usb-flash-drive-hw-trojan.html

USB

Much more awesome if you take it literally.

[12]

[12] Quelle: https://imgur.com/gallery/A6ZPPA1

[13]

[13] Quelle: https://twitter.com/manjusrii/status/455107685763911680/photo/1