

Entwicklung eines Treibers für einen Filmscanner mittels USB-Sniffing und Reverse Engineering

Hugo Platzer

Reflecta CrystalScan 7200

Zum Digitalisieren von Kleinbilddias / -negativen





- Per USB verbunden
- Software nur für Windows / Mac
- Versuch, einen Linux-Treiber zu entwickeln ohne entsprechende Dokumentation

Reverse Engineering

- Prozess des "Engineering" (von der Spezifikation zum Produkt) in umgekehrter Richtung
- Versuch, aus für den Konsumenten verfügbaren Daten (Firmware, Software, Mitlauschen an Schnittstellen)
 Protokolle zu rekonstruieren
- Rechtliche Grauzone
- Um Einverständnis gebeten: Hersteller war kooperativ

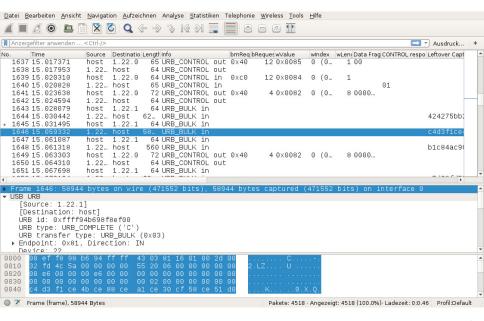
Der USB-Standard

- ein Anschluss für alle Klassen von Geräten
- Alle Kommunikation geht vom Host aus
- Ubertragung in Paketen > Transactions > Transfers
- ein Gerät hat oft mehrere Kanäle (Endpoints)
- Control-Endpoint
 - besitzt jedes Gerät, dient zur Steuerung
 - definierte Nachrichtenform mit Parametern und Zusatzdaten
 - Standard Requests: Vom USB-Standard vorgeschriebene Basisfunktionen
 - Vendor-specific Requests: Herstellerspezifische Zusatzfunktionen, undokumentiert
- Bulk-Endpoint
 - für große Datenmengen (Dateien, Bilddaten)
 - Bytestrom ohne Struktur

USB-Sniffing

- Mitschneiden der Kommunikation zwischen Gerät und Windows-Software
- Kernelmodul usbmon
 - Zugang zu vom Linux-Kernel abgearbeiteten USB-Transfers
 - Schwierigkeiten bei langer Payload
- Wireshark
 - Netzwerk-Sniffer, der sich auch für USB eignet
 - sowohl zur Aufzeichnung als auch Analyse
 - flexibel anpassbare Oberfläche
- VirtualBox
 - Windows in virtueller Maschine unter Linux ausführen
 - USB-Passthrough gibt Windows Zugang zum Scanner

Wireshark



usbmon: unvollständige Payload

- Bei großen Paketen wird Payload nach 61440 Bytes abgeschnitten
- visuell: Risse im rekonstruierten Bild



▶ gelöst durch einfachen Kernel-Patch
if (length >= rp->b_size / 5) length = rp->b_size / 5;

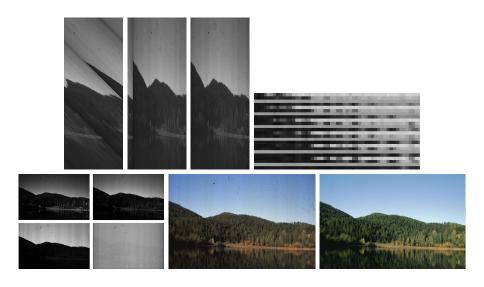
Analyse der Aufzeichnung

- Versuche, das übertragene Bild aus Aufzeichnung zu rekonstruieren
- ▶ Parsen des .pcap-File mit tshark
 - in Wireshark alle Protokolle außer USB deaktivieren!
- ▶ Bilddaten werden über große Bulk-Transfers übertragen
- zeilenweise 16-bit Little Endian Pixelwerte

```
40 c8 3b 5d 31 ae 2a 78 29 1c 29 bd 2e e9 33 93 40 50 21 43 25 2a 77 1d 76 1e 98 28 25 29 41 22 63 26 60 50 22 4a 1d 41 18 58 18 cf 17 9e 1a d3 1c 8b 1d 70 94 21 48 1e a8 24 10 1f 1f 15 78 15 15 1a 09 1b
```

- eingehende Bytes am Bulk-Endpoint sammeln, dann weiterverarbeiten:
 - Offsets
 - ▶ 16 → 8 Bit
 - In Zeilen gruppieren
 - usw.

Rekonstruktion des Bildes

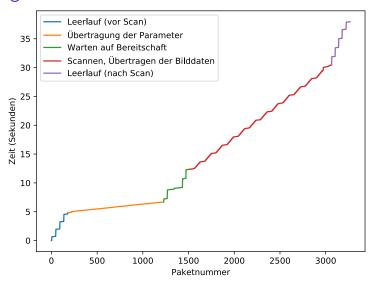


Ansteuern des Scanners

- PyUSB basiert
- ► Grundidee: aufgezeichnete Befehlsfolge zum Gerät wieder abspielen, eingehende Daten speichern
- viele Muster in der Kommunikation erkennbar

bmRec	bRe	w∨alue	W	nd	wLeng	Data Fragment	
0×40	12	0x0088	0		1	ff	
0×40	12	0x0088	0		1	aa	
0×40	12	0x0088	0		1	55	
0×40	12	0x0088	0		1	00	
0×40	12	0x0088	0		1	ff	
0×40	12	0x0088	0		1	87	
0×40	12	0x0088	0		1	78	
0×40	12	0x0088	0		1	e0	
0×40	12	0x0087	0		1	05	
0x40	12	0x0087	0		1	04	
0x40	12	0x0088	0		1	ff	
0×40	12	0x0085	0		1	dd	
0×40	12	0x0085	0		1	00	
0×40	12	0x0085	0		1	00	
0x40	12	0x0085	Θ		1	00	
0×40	12	0x0085	0		1	12	
0×40	12	0x0085	0		1	00	
0xc0	12	0x0084	0		1		
0×40	4	0x0082	0		8	0000000012000000	
0xc0	12	0x0084	0		1		
0xc0	12	0x0084	0		1		
0x40	12	0x0088	0		1	ff	
0×40	12	0x0088	0		1	aa	
0x40	12	0x0088	0		1	55	
0×40	12	0x0088	0		1	00	
0x40	12	0x0088	0		1	ff	

Zeitdiagramm



▶ an bestimmten Stellen auf Bereitschaft des Geräts warten, sonst Absturz

Funktion der jeweiligen Bytes

- ▶ Wie kann man ohne Dokumentation herausfinden, wie z.B. die Scanauflösung eingestellt werden kann?
- Zweimal scannen, nur diesen Parameter variieren
- ▶ Beispiel Auflösung, Abschnitt der übertragenen Bytefolge:
 - 300 dpi: 000f2c01802004000100000001801000
 1200 dpi: 000fb004802004000100000001801000
- nur wenige Parameter müssen verstanden werden, um das Gerät nutzbar zu machen

Bildverarbeitung

- Helligkeitsunterschiede pro Spalte ausgleichen
 - ▶ Helligkeit pro Spalte unterschiedlich
 - Scanner nimmt Leerbild auf
 - Ausgleichen: Division, nicht Subtraktion!



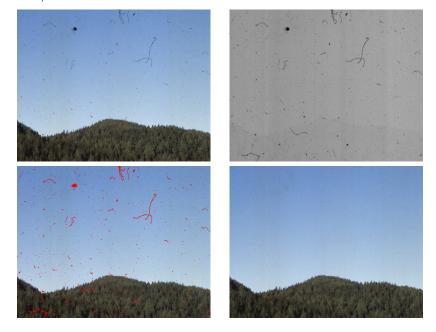






- Gammakorrektur, Negativmaske, Wertebereich
- automatisches Zuschneiden

${\sf Staub-}/{\sf Kratzerkorrektur}$



Ausblick: SANE

- Linux-Scannerframework mit GUI und Client/Server
- hat Treiber für CrystalScan 7200, funktioniert aber nicht bei diesem Exemplar
- zumindest Grundfunktionen (Scannen mit einstellbarer Auflösung) in SANE nutzbar machen

Live-Demo