GXS-700 X-ray Sensor
Part 2

John McMaster
JohnDMcMaster@gmail.com

Existing work

- Gave mtvre talk 3 years ago
- Made FOSS Gendex GXS-700 size 1 tools
- Same hardware in Dexis Platinum (right)

But wait! There's more

- Small size sensor for children ("size 2")
- Not supported since I didn't have a sensor

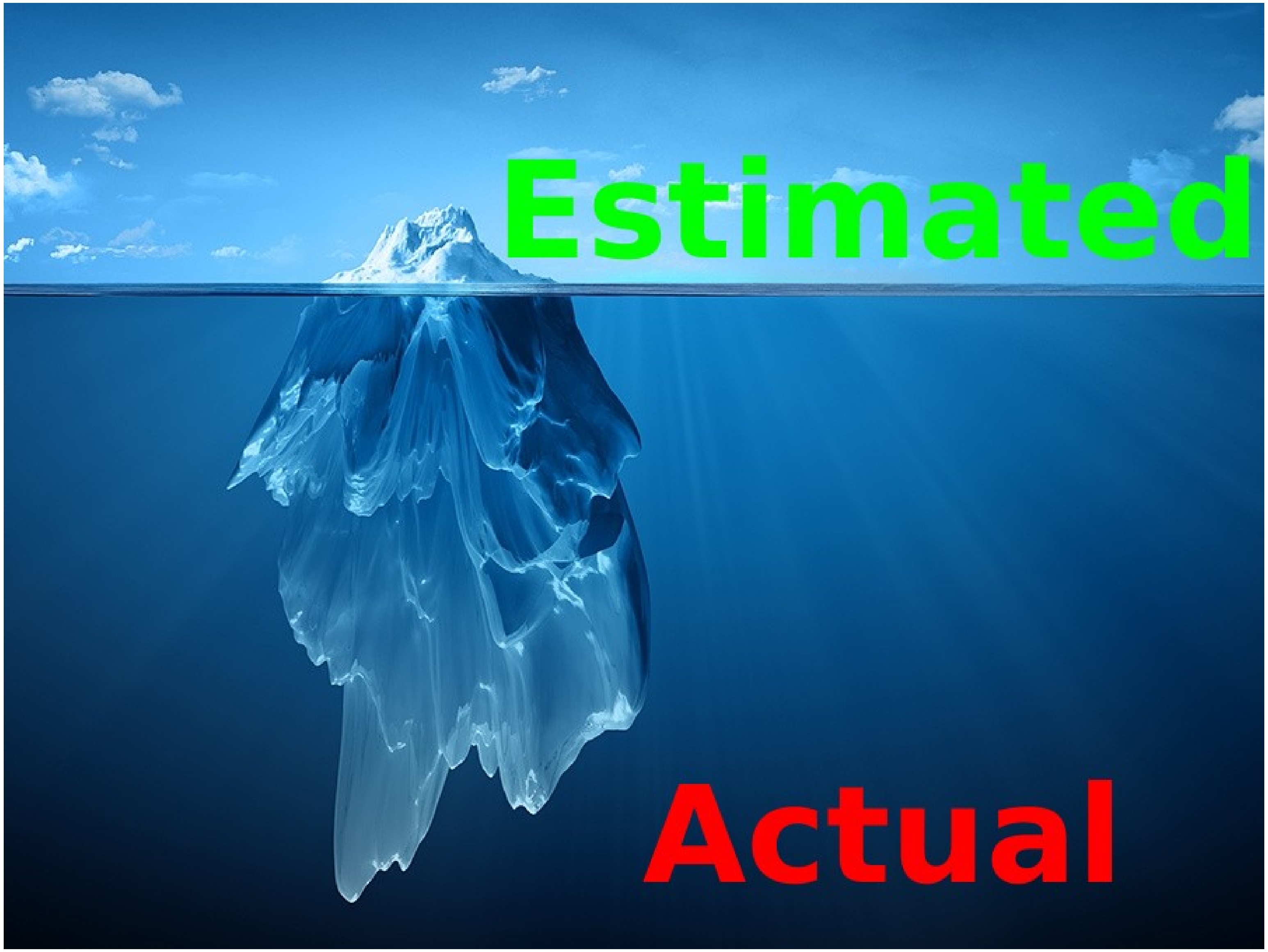
And then

- Customer wants size 2 support for project
- I say no since they want exclusive rights
- Eventually they agree to just support FOSS tool

A walk in the park?

- Should be easy project
- Already support closely related part
- Related capture, x-ray, etc infrastructure setup
- What could go wrong?

Project timeline

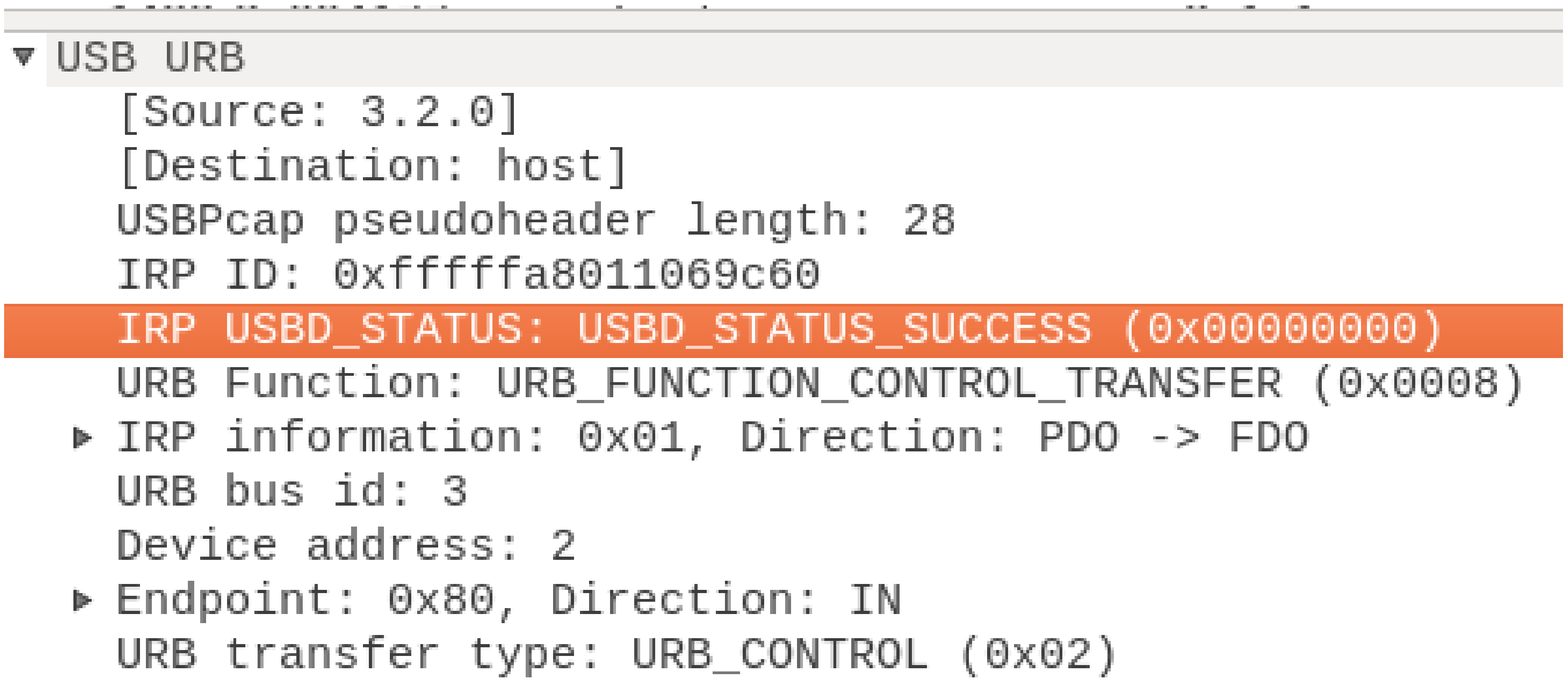


Getting samples

- Vendor reference image program not working
- Gendex / Dexis hack...possibly broke it?
- Sensor broken? They assured me no
- So had them capture samples, but...

Welcome to Windows

- Windows .pcap different than Linux .pcap
- Not as detailed
- Had to extend usbrply tool to support Windows



```
▼ USB URB
    [Source: 3.2.0]
    [Destination: host]
    USBPcap pseudoheader length: 28
    IRP ID: 0xfffffa8011069c60
    IRP USBD_STATUS: USBD_STATUS_SUCCESS (0x00000000)
    URB Function: URB_FUNCTION_CONTROL_TRANSFER (0x0008)
  ▶ IRP information: 0x01, Direction: PDO -> FDO
    URB bus id: 3
    Device address: 2
  ▶ Endpoint: 0x80, Direction: IN
    URB transfer type: URB_CONTROL (0x02)
```

Still not working...

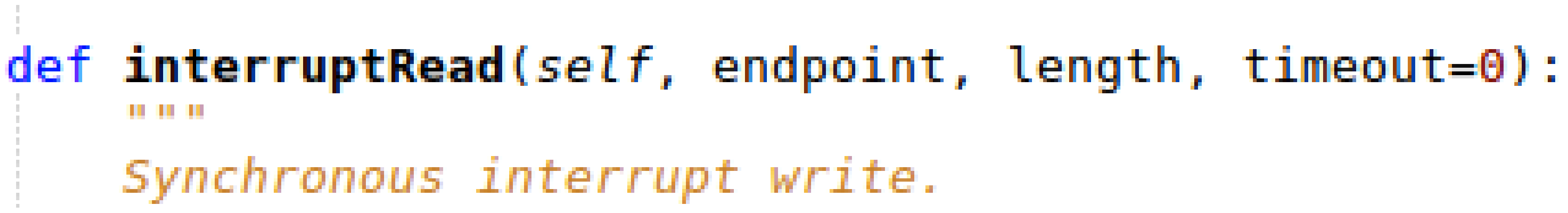
- Merged in changes from pcap, still doesn’t work
- Eventually realized large sensor uses USB bulk transfers, but small uses interrupt transfers!
- How did I miss this?
  - I wasn’t expecting this
  - Replay tool was set to omit these requests
  - Linux error messages were very generic

USB Interrupt Transfers

- Intended to be used for small ocassional events
- Primarily HID for keyboard / mouse
- Requires host to poll device
- Not inteded for large infrequent image transfers
- Why? Possibly small sensor doesn’t have proper image buffer?

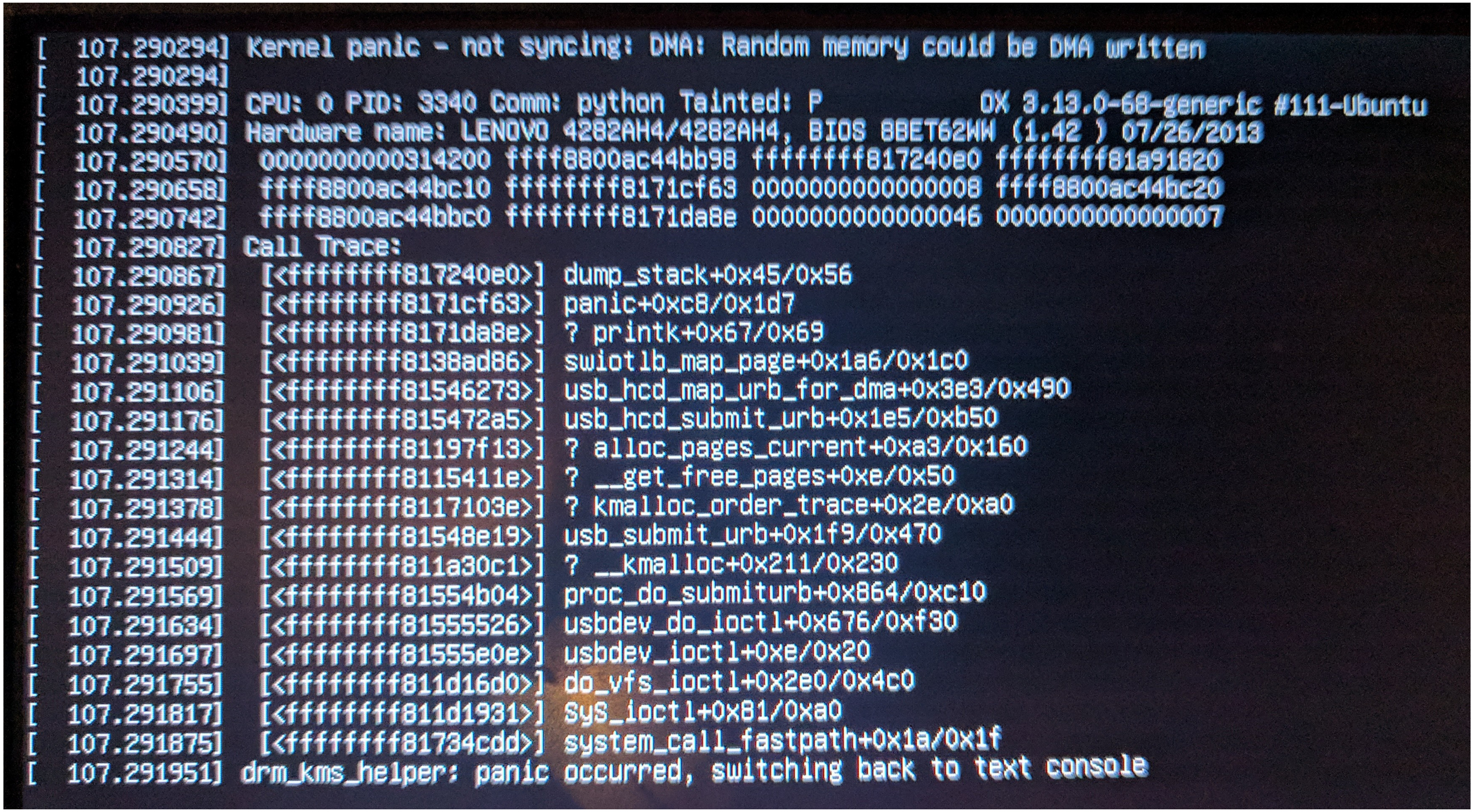
Things get worse

- I’m not familiar with USB Interrupt transfers
- Relatively poorly documented since rarely used
- Wasn’t exactly sure how to use the interface, but fiddled with it until it worked
- Then customer tried to use the code and...



```
def interruptRead(self, endpoint, length, timeout=0):
    """
    Synchronous interrupt write.
```

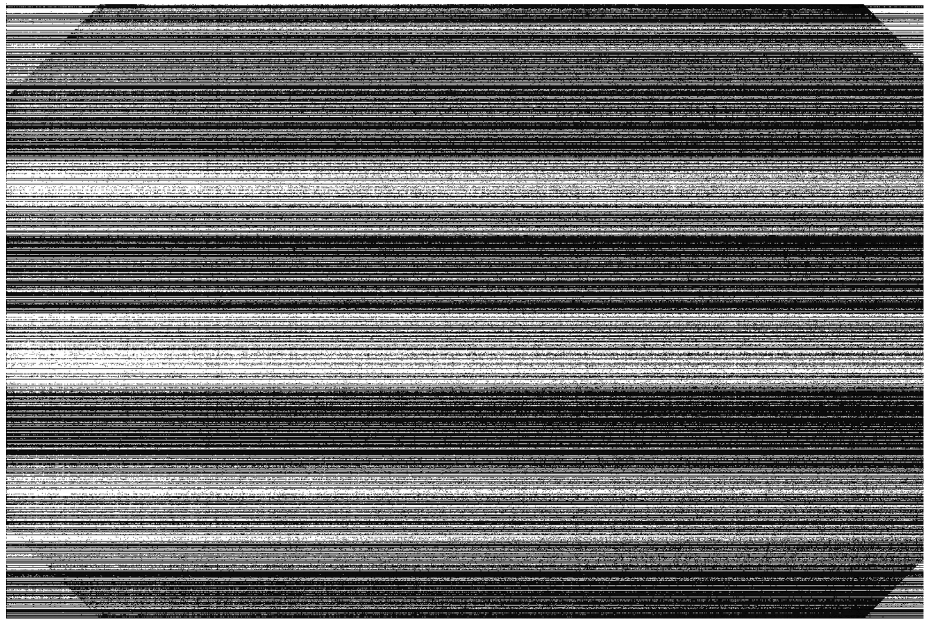
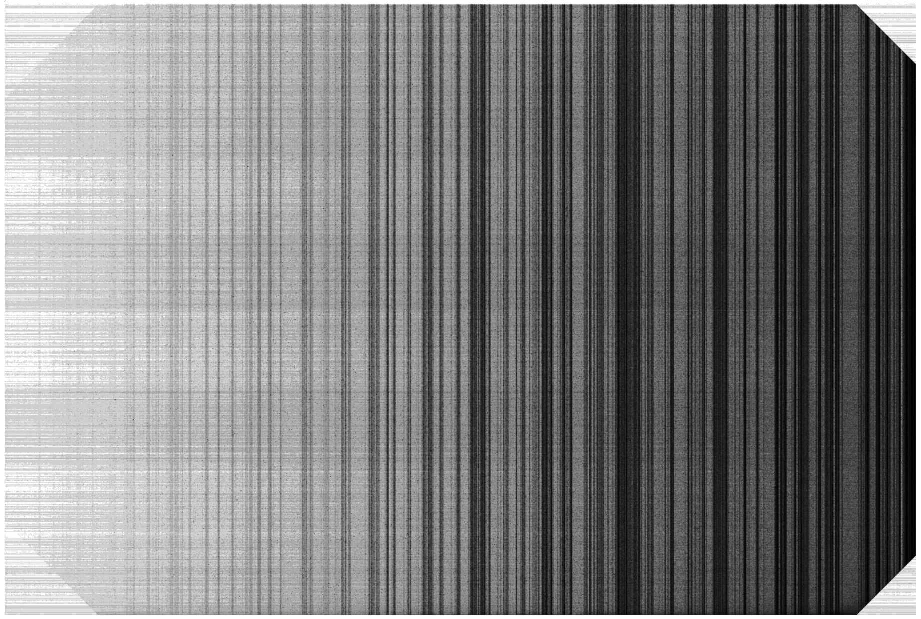
A slew of problems



```
[  107.290294] Kernel panic - not syncing: DMA: Random memory could be DMA written
[  107.290294]
[  107.290399] CPU: 0 PID: 3340 Comm: python Tainted: P           OX 3.13.0-68-generic #111-Ubuntu
[  107.290490] Hardware name: LENOVO 4282AH4/4282AH4, BIOS 8BET62WW (1.42 ) 07/26/2013
[  107.290570]  0000000000314200 ffff8800ac44bb98 ffffffff817240e0 ffffffff81a91820
[  107.290658]  ffff8800ac44bc10 ffffffff8171cf63 0000000000000008 ffff8800ac44bc20
[  107.290742]  ffff8800ac44bbc0 ffffffff8171da8e 0000000000000046 0000000000000007
[  107.290827] Call Trace:
[  107.290867]  [<ffffffff817240e0>] dump_stack+0x45/0x56
[  107.290926]  [<ffffffff8171cf63>] panic+0xc8/0x1d7
[  107.290981]  [<ffffffff8171da8e>] ? printk+0x67/0x69
[  107.291039]  [<ffffffff8138ad86>] swiotlb_map_page+0x1a6/0x1c0
[  107.291106]  [<ffffffff81546273>] usb_hcd_map_urb_for_dma+0x3e3/0x490
[  107.291176]  [<ffffffff815472a5>] usb_hcd_submit_urb+0x1e5/0xb50
[  107.291244]  [<ffffffff81197f13>] ? alloc_pages_current+0xa3/0x160
[  107.291314]  [<ffffffff8115411e>] ? __get_free_pages+0xe/0x50
[  107.291378]  [<ffffffff8117103e>] ? kmalloc_order_trace+0x2e/0xa0
[  107.291444]  [<ffffffff81548e19>] usb_submit_urb+0x1f9/0x470
[  107.291509]  [<ffffffff811a30c1>] ? __kmalloc+0x211/0x230
[  107.291569]  [<ffffffff81554b04>] proc_do_submiturb+0x864/0xc10
[  107.291634]  [<ffffffff81555526>] usbdev_do_ioctl+0x676/0xf30
[  107.291697]  [<ffffffff81555e0e>] usbdev_ioctl+0xe/0x20
[  107.291755]  [<ffffffff811d16d0>] do_vfs_ioctl+0x2e0/0x4c0
[  107.291817]  [<ffffffff811d1931>] SyS_ioctl+0x81/0xa0
[  107.291875]  [<ffffffff81734cdd>] system_call_fastpath+0x1a/0x1f
[  107.291951] drm_kms_helper: panic occurred, switching back to text console
```

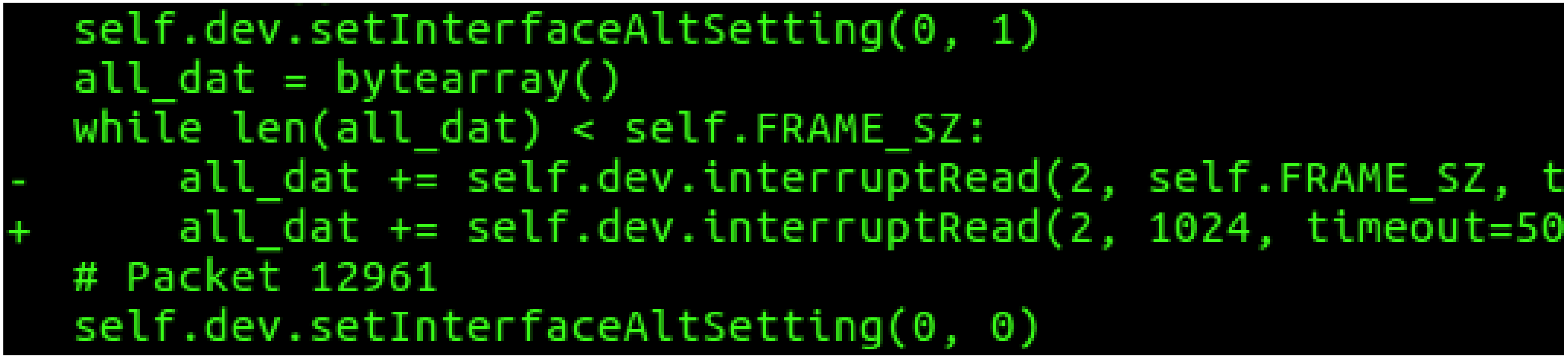
The world is on fire!

- Some native Linux: kernel panic
- VirtualBox: crashes Windows host
- Some systems yield corrupt image

What happened?

- Improper Interrupt transfer works on some systems
- Relatively recent kernels still had insufficient error checking
- Must keep alignment, even though over-reads device



```
   self.dev.setInterfaceAltSetting(0, 1)
   all_dat = bytearray()
   while len(all_dat) < self.FRAME_SZ:
-      all_dat += self.dev.interruptRead(2, self.FRAME_SZ, t
+      all_dat += self.dev.interruptRead(2, 1024, timeout=50
   # Packet 12961
   self.dev.setInterfaceAltSetting(0, 0)
```

Takeaways

- Bill by hour, not by project => $win
- Customer was happy despite under estimate due to clear communication
- Hard to estimate some projects up front
- Learned a little about Windows kernel USB
- Learned more about USB

Thanks!

- Questions? Interested?
  - JohnDMcMaster@gmail.com
- http://siliconpr0n.org/media/mtvre/2015-04-08_john_m_x-ray.pdf
- https://github.com/JohnDMcMaster/uvscada
- https://siliconpr0n.org/nuc/doku.php?id=gendex:gxs700