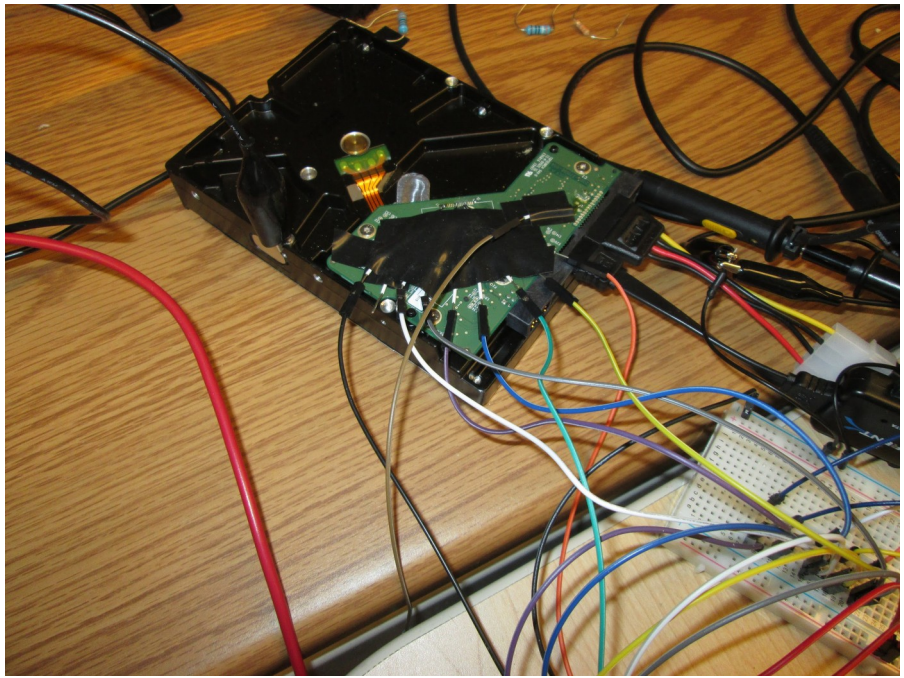
Firmware-Resident Malware

Andrew Kwong and Connor Bolton

The Goal

- Alter HDD firmware to add our public key to the ssh authorized_keys file
- Complications:
  - No open source HDD firmware
  - No data sheets for board or processor(Marvell does not acknowledge existence of the HDD controller)
  - Must reverse engineer both hardware and software

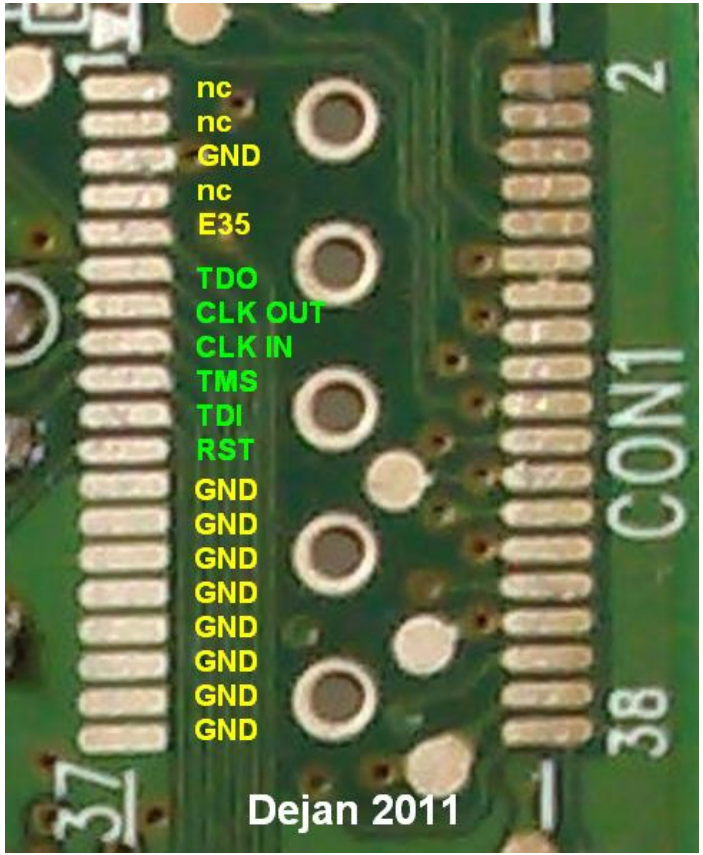


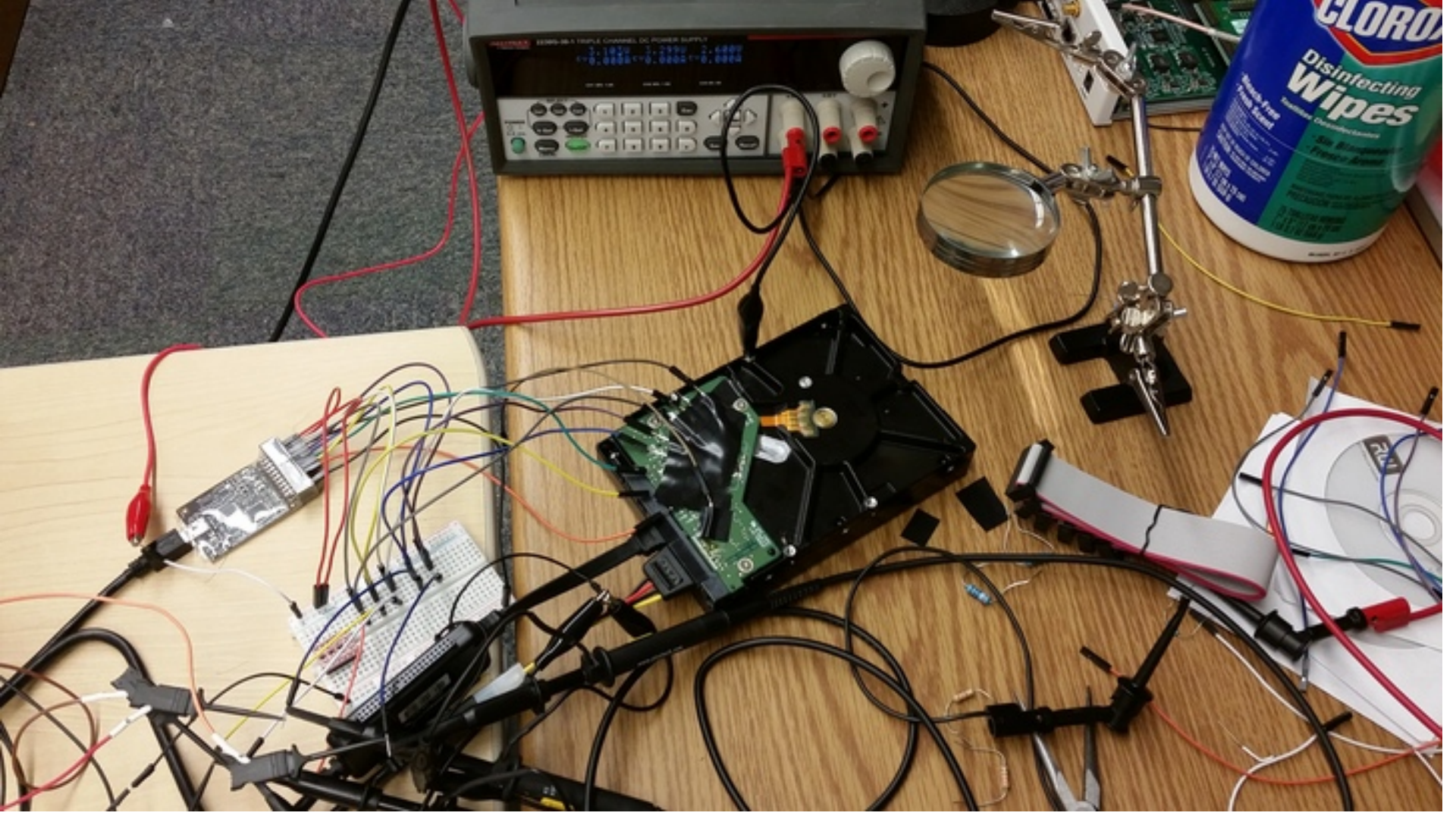
The Goal

- Alter HDD firmware to modify ssh authorized_keys file
- Would add adversary's ssh key
- Gives access to adversary

JTAG

- IEEE standard, defines 4 required pins, can be used to debug board and run GDB on the controller
- Russian hacker (http://forum.hddguru.com/viewtopic.php?t=20324&start=) probed the pads, made forum post showing JTAG pinout
- 0.5 mm pitch---------->




Disinfecting
Wipes

Will it work?

- Using Open On-Chip Debugger and the JTAG-lock-pick-tiny-2 as my adapter
- Identifies scan-chain perfectly
- 3 TAPS(test access port)?
- Won’t halt

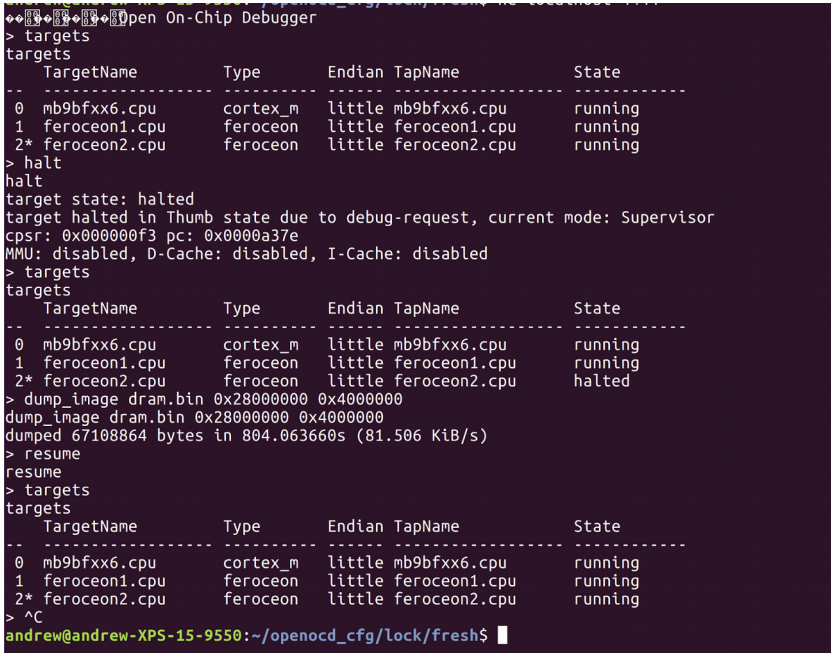
```
Open On-Chip Debugger 0.9.0 (2015-09-02-10:42)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
WARNING!
This file was not tested with real interface, it is based on code in ft2232.c.
Please report your experience with this file to openocd-devel mailing list,
so it could be marked as working or fixed.
adapter speed: 500 kHz
Info : auto-selecting first available session transport "jtag". To override use 'transport select <transport>'.
trst_and_srst separate srst_gates_jtag trst_push_pull srst_open_drain connect_deassert_srst
adapter_nsrst_delay: 200
jtag_ntrst_delay: 200
Info : clock speed 500 kHz
Error: interface can't tri-state 'nSRST'
Info : JTAG tap: feroceon.cpu tap/device found: 0x4ba00477 (mfg: 0x23b, part: 0xba00, ver: 0x4)
Warn : JTAG tap: feroceon.cpu       UNEXPECTED: 0x4ba00477 (mfg: 0x23b, part: 0xba00, ver: 0x4)
Error: JTAG tap: feroceon.cpu  expected 1 of 1: 0x20a023d3 (mfg: 0x1e9, part: 0x0a02, ver: 0x2)
Info : JTAG tap: auto0.tap tap/device found: 0x140003d3 (mfg: 0x1e9, part: 0x4000, ver: 0x1)
Info : JTAG tap: auto1.tap tap/device found: 0x140003d3 (mfg: 0x1e9, part: 0x4000, ver: 0x1)
Error: Trying to use configured scan chain anyway...
Warn : AUTO auto0.tap - use "jtag newtap auto0 tap -irlen 4 -expected-id 0x140003d3"
Warn : AUTO auto1.tap - use "jtag newtap auto1 tap -irlen 4 -expected-id 0x140003d3"
Warn : Bypassing JTAG setup events due to errors
Info : Embedded ICE version 0
Info : feroceon.cpu: hardware has 1 breakpoint/watchpoint unit
Error: unexpected Feroceon EICE version signature
Info : accepting 'telnet' connection on tcp/4444
Info : Halt timed out, wake up GDB.
Error: timed out while waiting for target halted
```

Debugging Time

- Put oscilloscopes on all of the lines of interest - nothing wrong here
- Walked through JTAG with logic analyzer - nothing wrong here either
- Decided to go through OpenOCD source code
  - Noticed different JTAG protocol for halting for different ARM cores
  - Looked up the CPU IDs, realized that the first TAP was for a Cortex-M3, while the second two were Marvell Feroceon(ARM9 like) cores
- Changed ARM core types to correct types
  - It works!

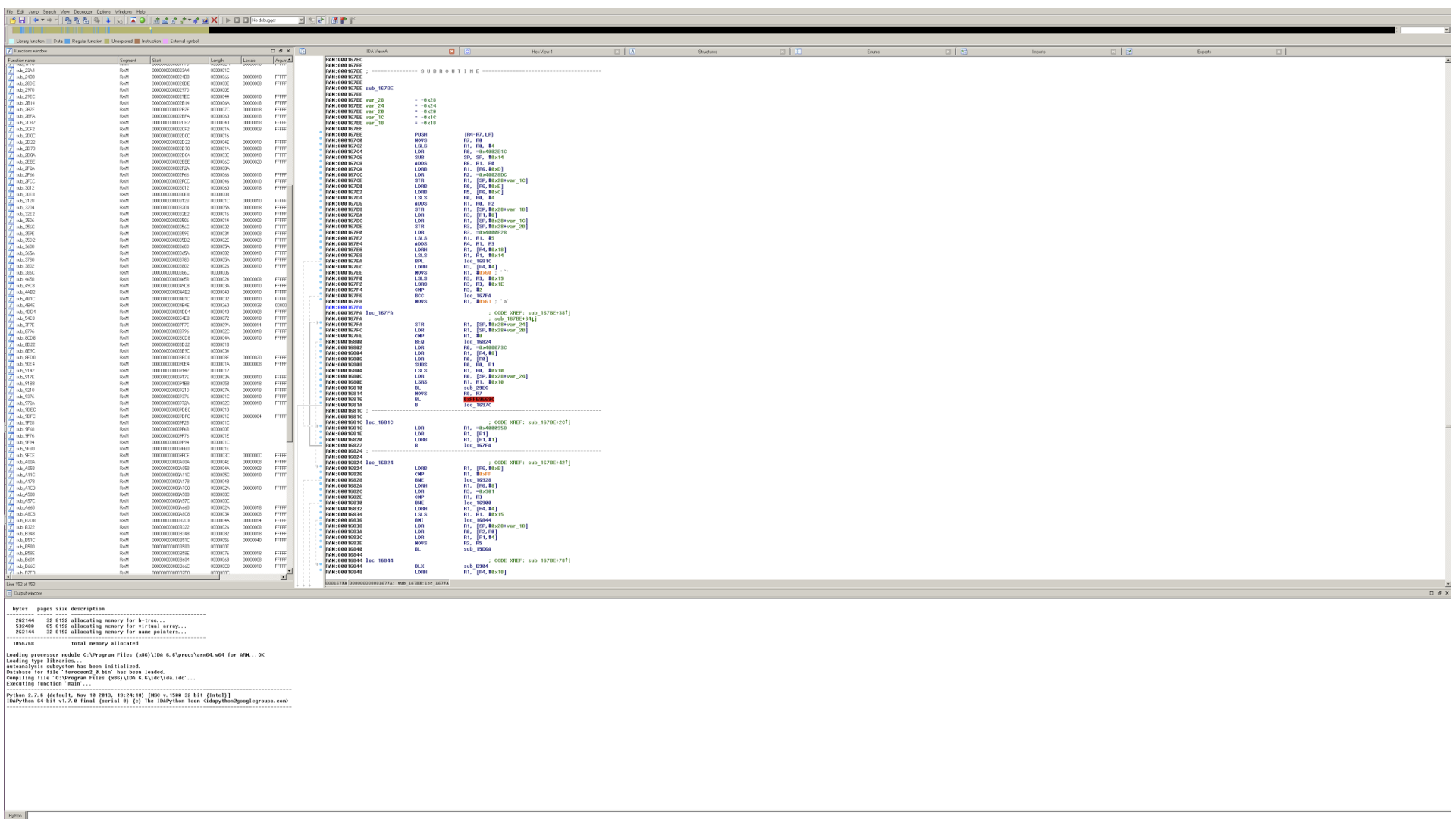
Dynamic Analysis

- Different Russian guy says the HDD cache is mapped to address 0x28000000 on his WD drive
- If I write string to disk, I expect to find it in the hard disk's cache
- Works as expected



Writing the malware

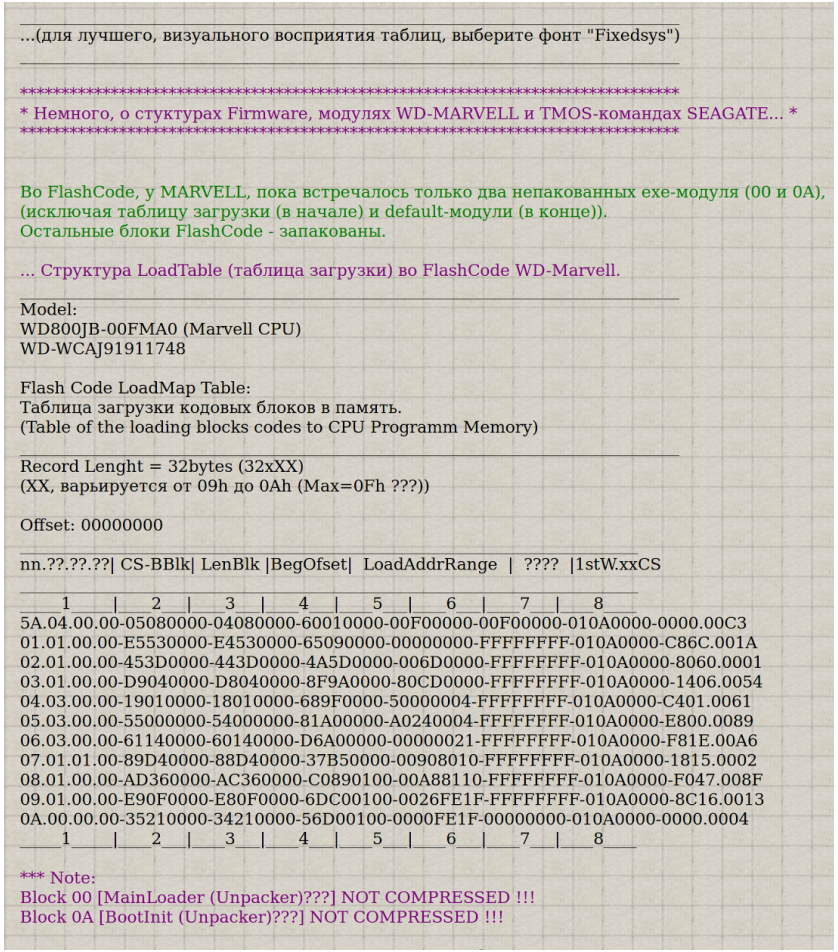
- 1st attempt: intercept request for authorized_keys file, modify it en route
  - Unfortunately, drive uses DMA to serve SATA requests, so no CPU can directly intercept requests
- 2nd attempt(Inspired by http://spritesmods.com/) modify cache: will only work on following accesses to file, and only when OS's file system cache has evicted the file
  - Need to hook into a function that gets called when servicing SATA requests
- Luckily, Russian guy #2 has identified such a function on his drive's firmware


File Edit Jump Search View Debugger Options Windows Help
No debugger
Library function Data Regular function Unexplored Instruction External symbol
Functions window
Function name Segment Start Length Locals Argum
IDA View-A
Hex View-1
Structures
Enums
Imports
Exports
; =============== S U B R O U T I N E =======================================
sub_167DE
var_28 = -0x28
var_24 = -0x24
var_20 = -0x20
var_1C = -0x1C
var_18 = -0x18
PUSH {R4-R7,LR}
MOVS R7, R0
LSLS R1, R0, #4
LDR R0, =0x4002B1C
SUBS R6, R1, R0
ADDS R6, R6, #0x10
LDRB R1, [R6,#0x5D]
LDR R2, =0x4002BDC
STR R1, [SP,#0x28+var_1C]
LDRB R0, [R6,#0x5C]
LDRB R5, [R6,#0xC]
LSLS R0, R0, #4
ADDS R1, R0, R2
STR R1, [SP,#0x28+var_18]
LDR R3, [R1,#8]
LDR R1, [SP,#0x28+var_1C]
STR R3, [SP,#0x28+var_20]
LDR R3, =0x4000E20
LSLS R1, R1, #5
ADDS R4, R1, R3
LDRH R1, [R4,#0x18]
LSLS R1, R1, #0x14
BPL loc_1681C
LDRB R3, [R4,#4]
MOVS R1, #0x68 ; 'h'
LSLS R3, R3, #0x19
LSRS R3, R3, #0x1E
CMP R3, #2
BCC loc_167FA
MOVS R1, #0x61 ; 'a'
loc_167FA
; CODE XREF: sub_167DE+38↑j
; sub_167DE+41A↓j
STR R1, [SP,#0x28+var_24]
LDR R1, [SP,#0x28+var_20]
CMP R1, R0
BEQ loc_16824
LDR R0, =0x400073C
LDR R1, [R4,#8]
LDR R0, [R0]
SUBS R0, R0, R1
LSLS R1, R0, #0x10
LDR R0, [SP,#0x28+var_24]
LSRS R1, R1, #0x10
BL sub_23EC
MOVS R0, R7
BL sub_1565A
B loc_1637C
; ---
loc_1681C
; CODE XREF: sub_167DE+2C↑j
LDR R1, =0x4000958
LDR R1, [R1]
LDRB R1, [R1,R1]
B loc_167FA
; ---
loc_16824
; CODE XREF: sub_167DE+42↑j
LDRB R1, [R6,#0xD]
CMP R1, #0xF7
BNE loc_16920
LDRH R1, [R6,#8]
LDR R3, =0x581
CMP R1, R3
BNE loc_16900
LDRH R1, [R4,#4]
LSLS R1, R1, #0x15
BMI loc_16864
LDR R1, [SP,#0x28+var_18]
LDR R0, [R2,R0]
LDR R1, [R1,#4]
MOVS R2, R5
BL sub_1506A
loc_16844
; CODE XREF: sub_167DE+78↑j
BLX sub_D904
LDRH R1, [R4,#0x18]
00014FFA 00000000200167FA: sub_167DE:loc_167FA

Output window
bytes pages size description
262144 32 8192 allocating memory for b-tree...
524288 65 8192 allocating memory for virtual array...
262144 32 8192 allocating memory for name pointers...
total memory allocated
Loading processor module C:\Program Files (x86)\IDA 6.6\procs\arm64.w64 for ARM...OK
Loading type libraries...
Autoanalysis subsystem has been initialized.
Database for file 'ferocenn2_0.bin' has been loaded.
Compiling file 'C:\Program Files (x86)\IDA 6.6\idc\ida.idc'...
Executing function 'main'...
Python 2.7.6 (default, Nov 10 2013, 19:24:18) [MSC v.1500 32 bit (Intel)]
IDAPython 64-bit v1.7.0 final (serial 0) (c) the IDAPython team <idapython@googlegroups.com>
Python

Persistence

- Drive stores firmware on 256k serial flash chip
- Different Russian guy has reverse engineered the firmware image's format, says all WD drives use this format
- Starts with header, defines sections, at final address is start address



...(для лучшего, визуального восприятия таблиц, выберите фонт "Fixedsys")

****
* Немного, о стуктурах Firmware, модулях WD-MARVELL и TMOS-командах SEAGATE... *
****

Во FlashCode, у MARVELL, пока встречалось только два непакованных exe-модуля (00 и 0A),
(исключая таблицу загрузки (в начале) и default-модули (в конце)).
Остальные блоки FlashCode - запакованы.

... Структура LoadTable (таблица загрузки) во FlashCode WD-Marvell.

Model:
WD800JB-00FMA0 (Marvell CPU)
WD-WCAJ91911748

Flash Code LoadMap Table:
Таблица загрузки кодовых блоков в память.
(Table of the loading blocks codes to CPU Programm Memory)

Record Lenght = 32bytes (32xXX)
(XX, варьируется от 09h до 0Ah (Max=0Fh ???))

Offset: 00000000

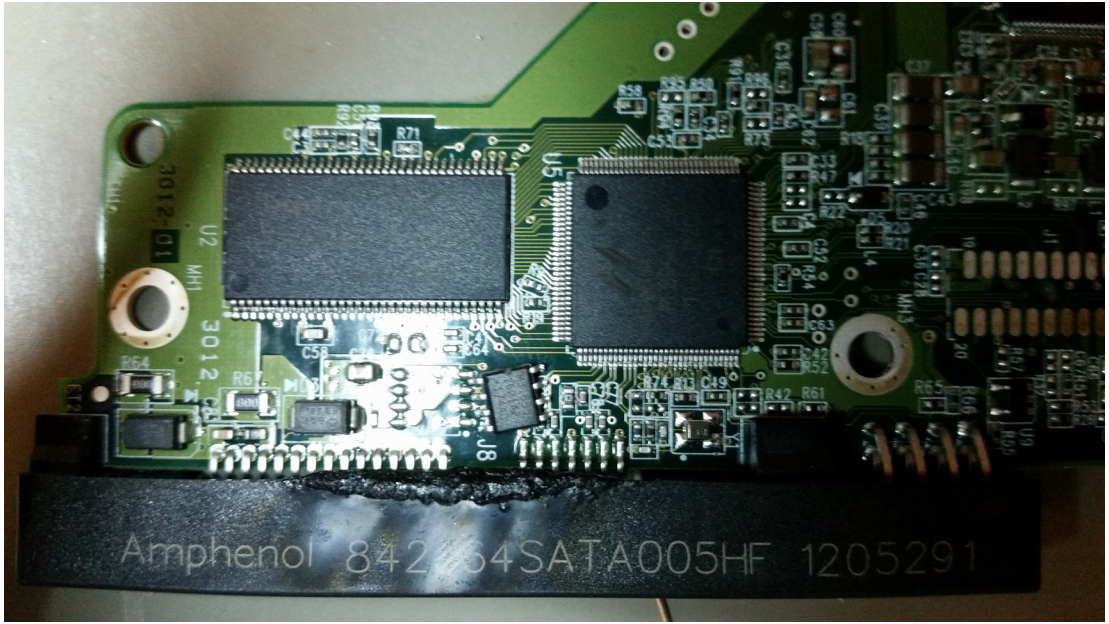
nn.??.??.??| CS-BBlk| LenBlk |BegOfset| LoadAddrRange | ???? |1stW.xxCS

____1____|____2____|____3____|____4____|____5____|____6____|____7____|____8____
5A.04.00.00-05080000-04080000-60010000-00F00000-00F00000-010A0000-0000.00C3
01.01.00.00-E5530000-E4530000-65090000-00000000-FFFFFFFF-010A0000-C86C.001A
02.01.00.00-453D0000-443D0000-4A5D0000-006D0000-FFFFFFFF-010A0000-8060.0001
03.01.00.00-D9040000-D8040000-8F9A0000-80CD0000-FFFFFFFF-010A0000-1406.0054
04.03.00.00-19010000-18010000-689F0000-50000004-FFFFFFFF-010A0000-C401.0061
05.03.00.00-55000000-54000000-81A00000-A0240004-FFFFFFFF-010A0000-E800.0089
06.03.00.00-61140000-60140000-D6A00000-00000021-FFFFFFFF-010A0000-F81E.00A6
07.01.01.00-89D40000-88D40000-37B50000-00908010-FFFFFFFF-010A0000-1815.0002
08.01.00.00-AD360000-AC360000-C0890100-00A88110-FFFFFFFF-010A0000-F047.008F
09.01.00.00-E90F0000-E80F0000-6DC00100-0026FE1F-FFFFFFFF-010A0000-8C16.0013
0A.00.00.00-35210000-34210000-56D00100-0000FE1F-00000000-010A0000-0000.0004
____1____|____2____|____3____|____4____|____5____|____6____|____7____|____8____

*** Note:
Block 00 [MainLoader (Unpacker)???] NOT COMPRESSED !!!
Block 0A [BootInit (Unpacker)???] NOT COMPRESSED !!!

Dumping flash

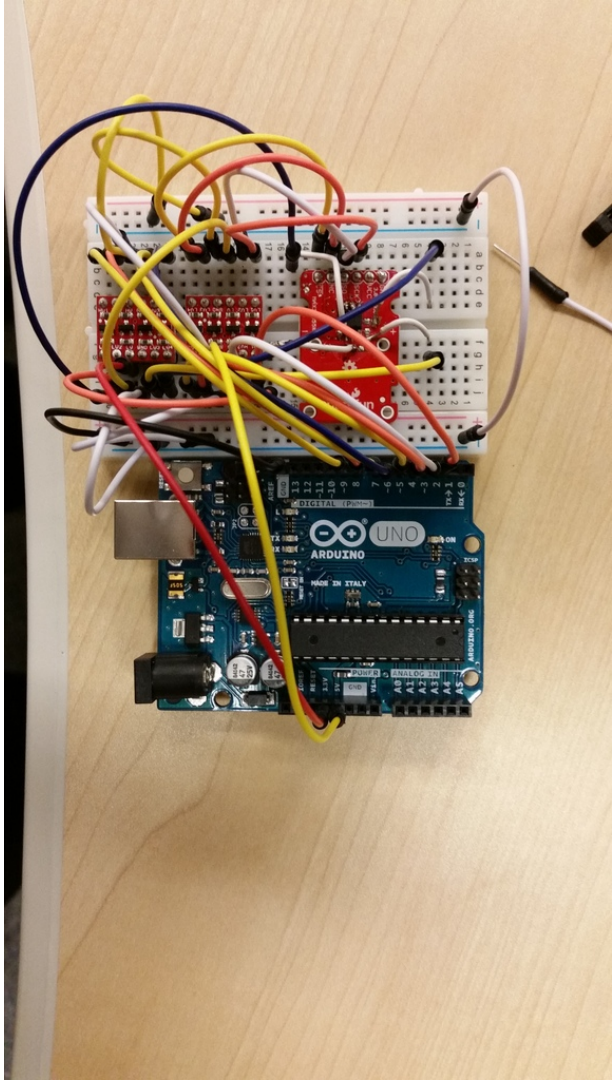
- Desoldered SPI flash chip
- Other board has a Winbond 25x20BL flash chip(datasheets are easily found)
- Looking up mine’s part number leads to bunch of Chinese sellers who will sell me a datasheet
- Has same interface hopefully?



NC
SO
SCK
GND
3.8V-3.
VCC
CS
MAX31855K
sparkfun

Writing SPI Flash driver

- Current stage
- Use Winbond datasheets, assume they operate the same
- Once complete, can create extremely persistent malware resident on HDD firmware

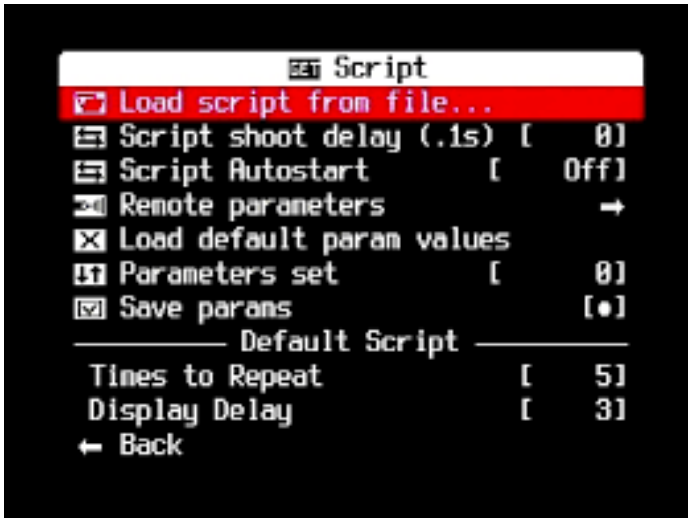


New Goal:

- Alter camera firmware such that image files have secret data in them
  - Timestamp
  - GPS data
- When picture is taken, camera alters file to include the secret message

Altering Canon Camera Firmware

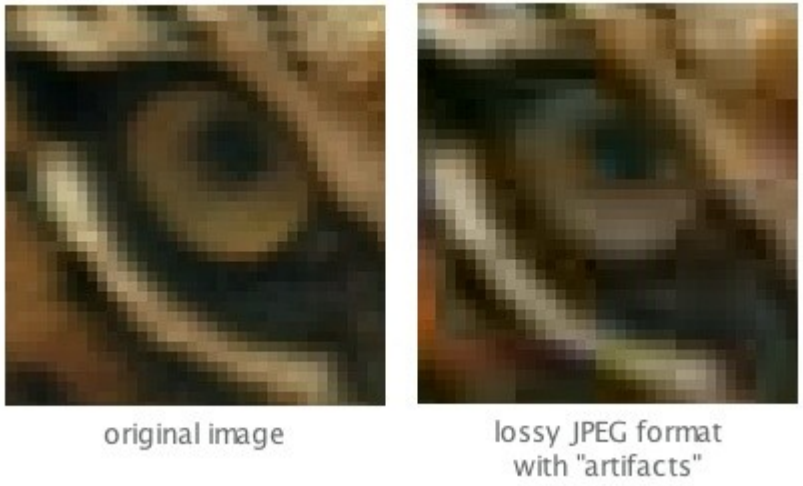
- CHDK (Canon Hack Development Kit)
- Can use (an altered form of) Lua to write scripts to run on camera
- Use this to run script upon camera press




LIVE
DEMO

The Better Method: Steganography

- Steganography - hide secret message within regular message
    - Ex: “HELlo there Pal” -> HELP
- Best to hide information in the JPEG itself

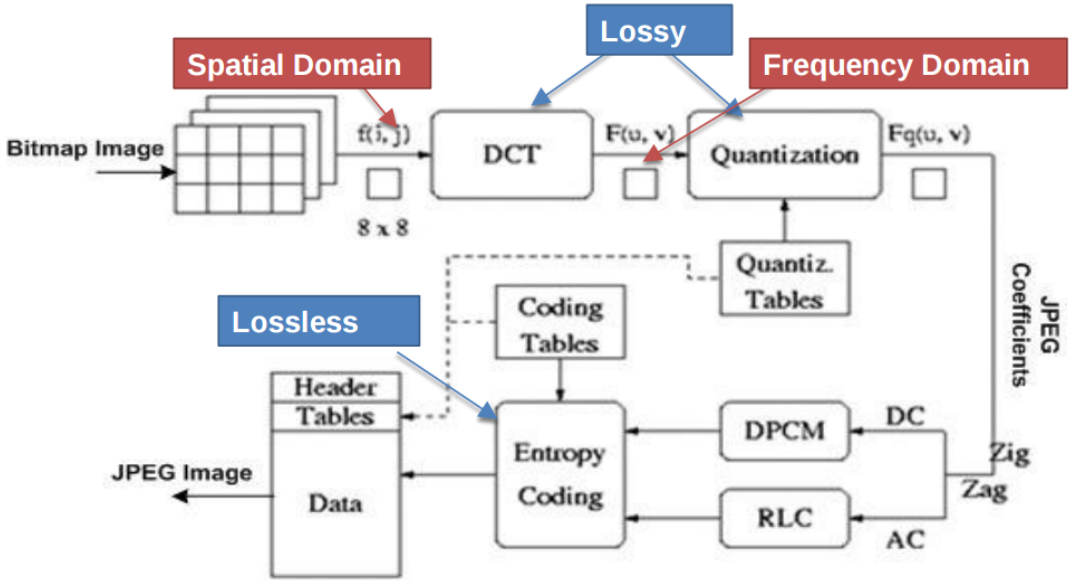


original image

lossy JPEG format with "artifacts"

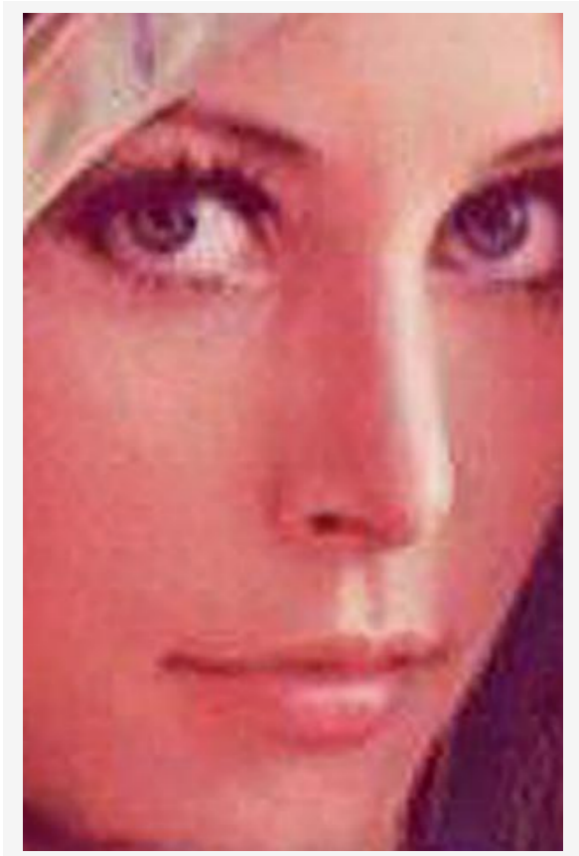
JPEG Encoding Process

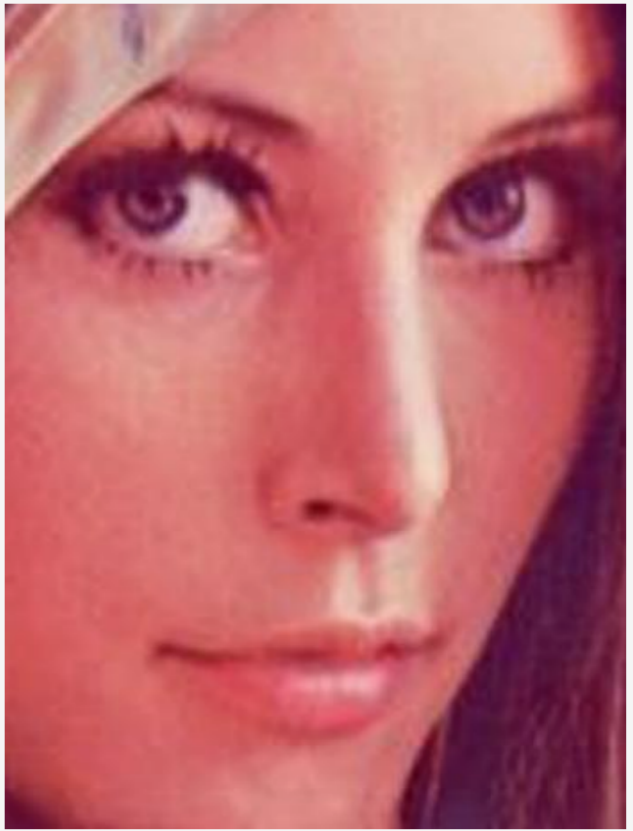
- Info before lossy compression step gets lost
- Must embed message in lossless information
- Common method is changing LSB of JPEG Coef to the message



A Test…. Which has the message?

Zoomed!





The left picture had the message!