

CRYPTOGRAPHY AND CAMERA HOMEBREW

Running code on ARM with lenses

TLP:GREEN

BIO

Laurent Clévy

- Forensic analyst in a SOC/CERT + some reverse engineering
- Following Magic Lantern (aka ML) activity since years
- Wanna be low level reverser
- Twitter: @lorenzo2472, Github: <https://github.com/lclevy>

Free time contributions

- Canon RAW v2 and RAW v3 file formats reference documentations
- Ported ML on 550d and 60d. Experiments on my R6
- Reversed Original Data Decision implementation from Canon in 2012, with python tool to recompute signatures. See this MISC article (open access):
<https://connect.ed-diamond.com/MISC/mischs-006/mecanisme-de-controle-d-authenticite-des-photographies-numeriques-dans-les-reflexes-canon>
- Other MISC MAG articles



DIGITAL CAMS ARE COMPUTING DEVICES

Digital Single Lens Reflex (DSLR) or Mirrorless Interchangeable Lens (MILC) cameras are complex devices

- Multiples CPUs (main, AF, peripherals, GPU, face recognition, network ops, ...)
- ARM-A9, ARM-M4 (mpu), Tensilica Xtensa (net), Takumi GV550 (gpu)
- RTOS (DryOS)
- Wifi, Bluetooth, Ethernet, GPS, USB, HDMI
- RAW image processing at 10-30 frame/sec

Solution : dedicated System on Chip for Canon, called DigIC (Digital IC)

So hackers want to run Doom on it !

<https://www.youtube.com/watch?v=fAoljXZYU7o> (Doom on RP by @coon)

<https://wiki.magiclantern.fm/digic> (which CPUs per Digic generation)



STANDING ON GIANTS SHOULDERS

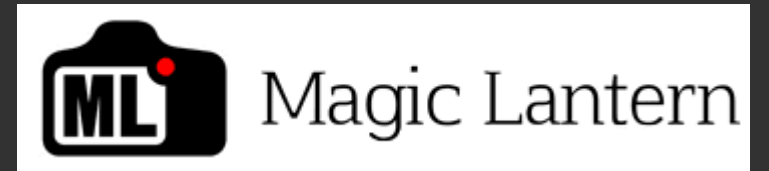
CHDK, Canon Hack DevKit (pocket cameras)

- <https://chdk.fandom.com/wiki/CHDK>
- Enhancing official firmware
 - RAW, LUA scripting,...
 - Loaded from sdcard, only in memory



Magic Lantern (DSLR)

- Created by Trammell Hudson (https://trmm.net/Magic_Lantern_firmware/)
 - Code execution in may 2009 on 5Dm2, with modified FIR update file
 - But blocked by signature check activation on 7D,
- 2010-2018 main contributors: A1ex (main dev), G3ggo (reverse), Arm.Indy (crypto), ...
- 2018-2022 : Kitor (reverse+dev), Names_are_hard, Coon, Petabyte, Turtius, ...



FIR FILES (VERSION 4, 2007-2018)

FIR header

- Camera model
- Update version
- FIR file checksum (sum of bytes)
- Pointers to other sections (updater₁, 2, 3, and fw records)
- Signature header (how to compute updaters signature)
- Updaters signature value (hmac-sha1)
- Firmware records signature value (hmac-sha1)

Updater₁ header

- Encryption header
- Encrypted payload (AES)

...

Firmware header

- Encrypted header
- Encrypted payload (AES)

See

- <https://groups.google.com/g/ml-devel/c/ASabsRbv9vQ/m/IKsMn3PPqPQJ?hl=en%7C7D> (Trammell, 10/2009)
- https://magiclantern.fandom.com/wiki/Firmware_file (Indy, 2011)

```
---.fir header---
0x000: modelId = 0x80000250, (7D, DryOS)
0x010: version = 1.0.9
0x020: checksum = 0x9e5642e5 integrity
0x024: updater1 header = 0xb0
0x028: updater1 offset = 0x120
0x02c: updater2 offset = 0x1c0970
0x030: firmware offset = 0x22e200
0x034: 0xffffffff
0x038: embedded file size = 0xc41b8c
0x03c: 0x0
0x040: sha1 seed = 0xb99b53de
0x044: 0x00000004 0x00000000 0x00000020 0x00000024 0x00000044 0x000000b0 0x0022e150 0x22e200 0xa1398c
0x068: updaters hmac-sha1 = ec50923d17342689204568e42d13a427d5462234 authentication
0x088: firmware hmac-sha1 = a34761111161b5125c1dfd3e7f3348a77a92f956
---updater1 header---
0x0b0: updater1 length = 0x1c0850. starts at 0x120
0x0b4: 0x1c0850
0x0b8: 0x0
0x0bc: xor seed value = 0x6dcb1922
0x120: --- updater1 (ciphered) --- confidentiality
0x1c0970: --- updater1 end ---
---updater2 header---
0x1c0970: (+0x000), modelId = 0x80000250, (7D, DryOS)
0x1c0980: (+0x010), version = 1.0.9
0x1c0990: (+0x020), checksum = 0xfd568edf
0x1c0994: (+0x024), 0xb0
0x1c0998: (+0x028), 0x120
0x1c099c: (+0x02c), ffffffff ffffffff ffffffff
0x1c09a8: (+0x038), updater length (including header) = 0x6d890. starts at 0x1c0970
0x1c0a20: (+0x0b0), updater length = 0x6d770. starts at 0x1c0a90
0x1c0a24: (+0x0b4), 0x6d764
0x1c0a28: (+0x0b8), 0x0
0x1c0b4c: (+0x0bc), xor seed value = 0x789a414c
0x1c0a90: (+0x120), --- updater2 (ciphered) ---
---firmware header---
0x22e200: (+0x000), offset to decryption data? = 0xc
0x22e204: (+0x004), offset to encrypted data = 0x7c. starts at 0x22e200
0x22e208: (+0x008), total firmware length (including header) = 0xa1398c. starts at 0x22e200
-
0x22e20c: (+0x00c), firmware length (encrypted part) = 0xa13910. starts at 0x7c
0x22e210: (+0x010), sha1_in = 0x0a1390e confidentiality
```

CODE EXECUTION : PATCHING UPDATER

40d (2007, VxWorks)

- Within update FIR file, updater code is **only xored with 2 tables** (512 and 513 bytes)
 - Updater is patched to dump memory (by ASalina, 2008)
 - Video hack based on liveview !
- Update records are AES encrypted

5dm2 (2008, DryOS)

- Bootflag activated by modified updater, then **autoexec.bin** loaded from sdcard
- Autoexec.bin : Canon feature likely for factory operations

7d (2009)

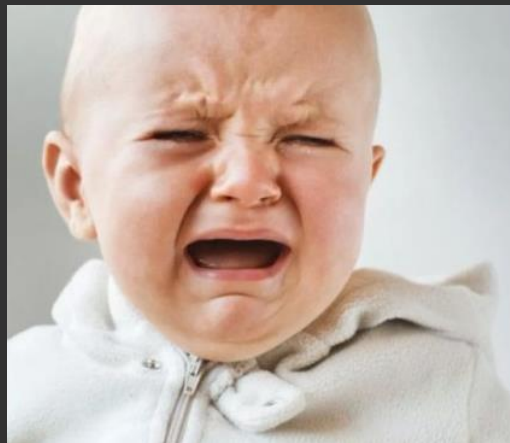
- **Signature check activated**. FIR Signature and decryption reversed (2010)
- ML team is able to forge FIR files (update files) like Canon, but Crypto tools kept private

550d (2010)

- Updater is now **encrypted with AES** too, and keys are changed

EOS R (09/2018) : CRYPTO CHANGED

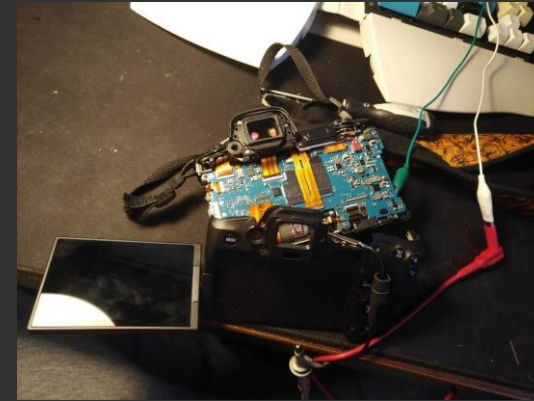
- FIR encryption changed, ML team can not decrypt Canon updates and reverse them for porting code
- FIR signature changed, no way to forge valid updates, run code to dump ROM
- No dumps, no reverse, no hacking !



EOS R, UART ACCESS

14/02/2019

- Kitor activated Bootflag via UART
- Custom autoexec.bin payload was used to dump
- Not suited to allow everyone to dump and port ML 😞



2/09/2020

- Free gift: **Canon basic** is available on EOS R !
- Discovered by Oren Isacson Alfredo Ortega, presented at DEFCON 18 (07/2010)
 - <https://www.youtube.com/watch?v=ByuHcamy2Z4>
- Known by CHDK people since 2010
 - <https://chdk.setepontos.com/index.php/topic,5549.o.html>
- Was not available on previous DSLRs, except m50 (04/2018)



CANON BASIC

Scripts can be executed under specific conditions

- See https://chdk.fandom.com/wiki/Canon_Basic

Scripts examples:

- Dump main ROM
- Enable Bootflag
- Dump camera log

```
private sub Initialize()  
    SaveAllRomImageToFile()  
end sub
```

```
EnableBootDisk()
```

```
dumpf()
```

- Allocate memory, load ARM code and jump to it with parameter !

- See https://github.com/lclevy/cbasic_examples/blob/main/R6.150/extend_cpuinfo.m

Format of the SD card

- The SD card may be in FAT16 or FAT32 format.
- The SD card must contain the following items:

1. The string "SCRIPT" must be at offset 0x1F0 of the first sector (Boot sector).
2. The file "script.req" must exist on the card's root directory, and must only contain the string "for DC_scriptdisk\n" (where the \n represents a newline character)
3. The file "extend.m" or "autotest.m" must exist on the root directory. This file must contain the Canon BASIC script to execute.

```
buf = umalloc(0x1000)  
if buf=0 then  
    exit sub  
end if  
bufo = umalloc(0x1000)  
if bufo=0 then  
    ufree(buf)  
    exit sub  
end if  
memset(bufo,0,0x1000)  
f = OpenFileRD(cpuinfo_binary)  
FIO_ReadFile(f,buf,0x1000)  
CloseFile(f)  
ExportToEventProcedure("cpuinfo",buf|1)  
cpuinfo(bufo)
```

PTP CANON EVENT PROCEDURES

Event procedures are “call by name” functions, works with PTP

- Available from Canon Basic
- From DryOS, ARM code: call(‘enablebootdisk’);
- From Picture Transfer Protocol, via a custom command : 0x9052

Petabyte (June 2021)

- Enabling bootflag using PTP on 1300D (digic4+)
- Reference client implementation is Canon EDSDK.DLL (from EOS UTILITY)

SRSA (06/02/2022)

- Describe **EventProcedure call encoding** with parameters
- See <https://chdk.setepontos.com/index.php?topic=4338.msg147738#msg147738>

```
Leftover Capture Data: 16000000020052900200000064756d70660000000000
▼ PTP/USB: Data
  Type: 2
  Transaction ID Offset: 8
  PTP Packet Length: 22
  Transaction ID: 0x00000002
  Code: 0x9052
  Packet Type in PTP/IP: 0x00000009
> MTP Packet:
0000 00 0c 9e 92 2d 89 ff ff 53 03 02 21 01 00 2d 00 .....S!....
0010 87 c3 7e 62 00 00 00 00 60 6e 02 00 8d ff ff ff ..~b....`n....
0020 16 00 00 00 16 00 00 00 00 00 00 00 00 00 00 00 .....
0030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0040 16 00 00 00 02 00 52 90 02 00 00 00 64 75 6d 70 .....R....dump
0050 66 00 00 00 00 00
```

CODE EXEC PATH PER DIGIC GENERATIONS

Code exec method	40D 2007 (d3)	5Dm2 2008 (d4)	1300D 2016 (d4+)	5D3 2012 (d5)	80D 2016 (d6)	5Dm4 2016 (d6+)	6D2 2017 (d7)	M50 04/2018 (d8)	EOS R 09/2018 (d8)	R6 2020 (dx)	R10 2022 (dx)
Cbasic	no	no	no	no	no	no	no	ok	ok	ok	locked
Fir v4	yes	yes	yes	yes	yes	yes	yes	yes	no, v5	no, v6	?
autoexec	?	yes	yes	yes	yes	yes	yes	yes	yes	yes	?
ptp	no?	yes	yes	yes	yes	yes	yes	yes	locked	locked	locked
uart	no?	?	ok	ok	ok	yes	ok	ok	ok	ok	locked

Canon needs code execution in factories to personalize / calibrate cameras

- Cbasic, autoexec and PTP features do certainly exist for that

Imagination (and time) are the limits ...

OTHER REFERENCES

EOS 300d decryption, Alex Bernstein, 2003
cpu=x86, mpu=mips, os=DOS

<https://web.archive.org/web/20100814164946/wiki.alexbernstein.com/FirmwareDecrypterUnpacker>

Homebrew firmware for the 5D Mark 2, Trammell Hudson (17/05/2009)

<https://www.dvinfo.net/forum/canon-eos-full-frame-hd/235535-homebrew-firmware-5d-mark-2-a.html>

ML running on 550d , Trammell Hudson (31/07/2010)

<https://www.dvinfo.net/forum/canon-eos-crop-sensor-hd/482742-magic-lantern-demo-550d.html>

Magic Lantern Free Software on your camera, Libre Graphics Meeting, Leipzig, 2/04/2014,

<https://libregraphicsmeeting.org/2014/slides/magic.lantern.libgre.graphics.meet.2014.pdf>

https://download.gimp.org/pub/gimp/lgm/2014/day_1/017_Magic_Lantern_Free_Software_on_Your_Camera/017_Magic_Lantern_Free_Software_on_Your_Camera.ogv

EOS firmware in QEMU - development and reverse engineering guide, Alex

<https://foss.heptapod.net/magic-lantern/magic-lantern/-/blob/branch/qemu/contrib/qemu/HACKING.rst>

ML discord server : <https://discord.gg/TyRPtvAS>