Exiv2 - Bug #1105

exiv2 output is inconsistent and seemingly random 1% of the time

13 Aug 2015 00:33 - Daniel Lu

Status:	Closed	**Start date:**	13 Aug 2015
Priority:	Normal	**Due date:**	
Assignee:	Robin Mills	**% Done:**	100%
Category:	not-a-bug	**Estimated time:**	4.00 hours
Target version:	0.26		

Description

I ran the following script:

```
#!/bin/bash
for i in `seq 1000`
do
    exiv2 -q DSC01825.jpg | grep 'Image timestamp' >> test2.txt
end
```

The resulting 1000 lines contains mostly "Image timestamp" with the correct timestamp as desired, but a few lines say instead "Binary file (standard input) matches" (test2.txt attached). This indicates that exiv2 is occasionally outputting certain characters causing grep to think it is a Binary file. But I could see no pattern to this, suggesting that exiv2 is behaving in a flaky way.

Also, it can't possibly be that my hardware is failing because I also ran sha1sum 1000 times and the output was the same all 1000 times (test3.txt attached).

I've also uploaded the picture. (DSC01825.jpg attached)

Using exiv2 0.25 001900 (64 bit build) on Arch Linux (kernel version 4.1.4).

The picture was taken with a Sony ILCE-7R.

The expected output of exiv2 should be consistent and not randomly change about 1% of the time.

Related issues:

Related to Exiv2 - Bug #740: Error: Offset of directory Sony1, entry 0x2001 i...	**Closed**	**12 Nov 2010**
Related to Exiv2 - Feature #1108: Recursively dump sub-files of an image	**Closed**	**21 Aug 2015**

History

#1 - 13 Aug 2015 18:13 - Robin Mills

Nice photo. Seattle, I think.

I can't reproduce this on MacOS-X 10.10 (Yosemite), Cygwin, or Linux Ubunutu 15.04 with Exiv2 v0.25 svn=3883 when I use this image: http://dev.exiv2.org/attachments/download/805/DSC_7154.jpg

I can't reproduce this with your file and script on MacOS-X. I can reproduce this with your file on Ubunutu 15.04 and Cygwin. However your test file is suspect:

```
$ for i in {1..300}; do exiv2 -pa -g DateTimeOriginal DSC01825.jpg ; done
Error: Offset of directory Sony1, entry 0x2001 is out of bounds: Offset = 0x00901076; truncating the entry
Exif.Photo.DateTimeOriginal                  Ascii      20  2015:07:09 00:47:56
.....
$
```

I don't know what the meaning of the message *"Error: Offset of directory...."*.

I've dumped the structure of the file without discovered anything about directory Sony1.

```
261 rmills@rmillsmbp-k1504:~/temp/foo $ exiv2 -pS DSC01825.jpg
STRUCTURE OF JPEG FILE: DSC01825.jpg
 address | marker     | length  | data
       2 | 0xd8 SOI   |       0 |
       4 | 0xe1 APP1  |   48842 | Exif..II*.......... ..........
   48848 | 0xe2 APP2  |     304 | MPF.II*.............0100.....
   49154 | 0xdb DQT   |     132 |
```

```
   49288 | 0xc4 DHT   |     418
   49708 | 0xc0 SOF0  |      17
   49727 | 0xda SOS   |      12
$
```

It's odd that I only get the message on Linux and Cygwin - however both use the GCC compiler. The Mac compiles with clang. I've built exiv2 on Linux with clang and that reproduces your fault. So although this issue can only be reproduced on some platforms, it doesn't appear to be compiler dependent.

Its unlikely to be a coincidence that the message and the random binary output arrive together on the same platform. It almost feels like a buffer is not being correctly filled.
I don't see any evidence that exiv2 is not consistent with good files. The exiv2 test suite would not work if we were producing random results on solid test files.

However there seems to be something odd about your test file DSC01825.jpg. I'm not going to pursue that investigation at this time in the hope that somebody else can explain the message. This matter was reported several years ago #740.

#2 - 13 Aug 2015 19:13 - Robin Mills

Solved!

I've made a couple of interesting discoveries about this:

1) Meaning of message *Offset of directory ...*
http://sourceforge.net/p/darktable/mailman/message/29000541/

2) This is only related to the output from exiv2's default output. (e.g. exiv2 DSC01825.jpg) sometimes producing binary output.

If I run these commands:

```
$ exiv2 -pa -q -g Original DSC01825.jpg;
Exif.Photo.DateTimeOriginal                  Ascii      20  2015:07:09 00:47:56
$ for i in {1..3000}; do exiv2 -pa -q -g Original DSC01825.jpg; done | wc
   3000   15000  240000
$
```

I've trapped the culprit with this command:

```
$ exiv2 -q DSC01825.jpg | nl
     1  File name       : DSC01825.jpg
     2  File size       : 10125312 Bytes
     3  MIME type       : image/jpeg
     4  Image size      : 7360 x 4912
     5  Camera make     : SONY
     6  Camera model    : ILCE-7R
     7  Image timestamp : 2015:07:09 00:47:56
     8  Image number    :
     9  Exposure time   : 1/10 s
    10  Aperture        : F2.8
    11  Exposure bias   : 0 EV
    12  Flash           : No, compulsory
    13  Flash bias      : 0 EV
    14  Focal length    : 28.0 mm (35 mm equivalent: 28.0 mm)
    15  Subject distance:
    16  ISO speed       : 800
    17  Exposure mode   : Aperture priority
    18  Metering mode   : Multi-segment
    19  Macro mode      :
    20  Image quality   : n/a
    21  Exif Resolution : 7360 x 4912
    22  White balance   : Auto
    23  Thumbnail       : image/jpeg, 10408 Bytes
    24  Copyright       :
    25  Exif comment    :
```

```
$ for i in {1..100};do exiv2 -q DSC01825.jpg | head -24| grep timestamp ; done
Image timestamp : 2015:07:09 00:47:56
Image timestamp : 2015:07:09 00:47:56
...
```

There's something binary being written out occasionally in the Exif comment: field. And when I dump it:

```
$ exiv2 -q DSC01825.jpg | tail -2 | od -a
0000000   E   x   i   f  sp   c   o   m   m   e   n   t  sp  sp  sp  sp
0000020   :  sp nul nul nul nul nul nul nul nul nul nul nul nul nul nul
```

```
0000040 nul nul nul nul nul nul nul nul nul nul nul nul nul nul nul nul
*
0000100 nul nul nul nul nul nul nul nul nul nul  nl  nl
0000114
$
```

Indeed there are binary bytes! Why grep is being inconsistent in detecting these binary bytes I do not know. exiv2 is consistent. grep is not. Case Closed.

I'll update #740 with the explanation about the error message: http://sourceforge.net/p/darktable/mailman/message/29000541/

#3 - 13 Aug 2015 20:06 - Daniel Lu

Thanks for investigating. It is odd that GNU grep would behave this way, but it kind of makes sense if the input is piped through stdin. It could be the case that, if exiv2 is running slowly, then grep optimistically prints out matching lines if the input lines are coming in slow enough that grep matches the "Image timestamp" line before receiving the binary bytes at the end. But on fast days, grep receives all the lines at once and summarily decides the whole thing is binary. Not too sure...

I also tried:

```
exiv2 -q DSC01825.jpg > exif.txt
for i in {1..100}
do
    grep 'Image timestamp' exif.txt >> test4.txt
end
for i in {1..100}
do
    cat exif.txt | grep 'Image timestamp' >> test5.txt
end
```

and the output is consistent in both test4 and test5: they are all "Binary file (standard input) matches".

In any case, as a design decision: maybe exiv2 should not print out binary data at all, should there be any binary data present in the EXIF comment. Imagemagick, for example, prints out the byte values as a list of integers between 0 and 255.

```
identify -verbose DSC01825.jpg
```

By the way, DSC01825.jpg was (as far as I recall) unmodified straight out of camera. Although Sony does not seem to perfectly adhere to EXIF standards, perhaps exiv2 should behave nicer with it.

And you were right, the photo was taken in Seattle! It's Convention Place Station, actually.

#4 - 13 Aug 2015 21:21 - Robin Mills

Daniel

Right. I think we've just about done this to death. I suspect the grep business has something to do with flushing the pipe - however I'm not going to worry about it. The Mac provides BSD grep:

```
521 rmills@rmillsmbp:~/gnu/exiv2/trunk $ grep --version
grep (BSD grep) 2.5.1-FreeBSD
522 rmills@rmillsmbp:~/gnu/exiv2/trunk $
```

As you've said, Linux provides GNU grep.

```
314 rmills@rmillsmbp-k1504:~ $ grep --version
grep (GNU grep) 2.20
Copyright © 2014 Free Software Foundation, Inc.
Licence GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Written by Mike Haertel and others, see <http://git.sv.gnu.org/cgit/grep.git/tree/AUTHORS>.
315 rmills@rmillsmbp-k1504:~ $
```

I'm sure the code for GNU grep will build on the Mac - however I can't be bothered (it's 22:19 in England and I'm off to bed).

I've now retired and decided to leave Team Exiv2 in May (after 7 years of working on the project). So it's up to somebody else to decide if exiv2 should change behaviour. I'm a little bored this evening and thought your puzzle was interesting. Having satisfied my curiosity, I'm not going to get involved with any consequential tasks.

I've been to Seattle many times. I worked for Adobe and lived in San Jose, CA for 14 years. It's nice to be home in England. I miss the weather, but not the drought.

#5 - 14 Aug 2015 01:44 - Andreas Huggel

Great investigation, Robin, thanks!

#6 - 14 Aug 2015 11:25 - Robin Mills

Thanks, Andreas. I've made another amazing discovery about this, thanks to a comment by Jeroen in this thread:
http://dev.exiv2.org/boards/3/topics/1131

The "improper" data in the Sony1 image is the preview. Here's the proof:

1 Dump the Structure of DSC01825.jpg

```
561 rmills@rmillsmbp:~/temp/foo $ exiv2 -pS DSC01825.jpg
STRUCTURE OF JPEG FILE: DSC01825.jpg
 address | marker     | length  | data
       2 | 0xd8 SOI   |       0
       4 | 0xe1 APP1  |   48842 | Exif..II*.......... ..........
   48848 | 0xe2 APP2  |     304 | MPF.II*.............0100.....
   49154 | 0xdb DQT   |     132
   49288 | 0xc4 DHT   |     418
   49708 | 0xc0 SOF0  |      17
   49727 | 0xda SOS   |      12
$
```

2 Extract the APP1 segment into buff.tif and dump that:

```
562 rmills@rmillsmbp:~/temp/foo $ dd bs=1 skip=12 count=48842 if=DSC01825.jpg of=buff.tif ; exiv2 -pS buff.tif
48842+0 records in
48842+0 records out
48842 bytes transferred in 0.125983 secs (387687 bytes/sec)
STRUCTURE OF TIFF FILE (II): buff.tif
 address |    tag                          |      type |    count |   offset | value
      10 | 0x010e ImageDescription         |     ASCII |       32 |      158 |

      22 | 0x010f Make                     |     ASCII |        5 |      190 | SONY
      34 | 0x0110 Model                    |     ASCII |        8 |      196 | ILCE-7R
      46 | 0x0112 Orientation              |     SHORT |        1 |        1 | 1
      58 | 0x011a XResolution              |  RATIONAL |        1 |      204 | 204/0
      70 | 0x011b YResolution              |  RATIONAL |        1 |      212 | 212/0
      82 | 0x0128 ResolutionUnit           |     SHORT |        1 |        2 | 2
      94 | 0x0131 Software                 |     ASCII |       14 |      220 | ILCE-7R v2.00
     106 | 0x0132 DateTime                 |     ASCII |       20 |      234 | 2015:07:09 00:47:56
     118 | 0x0213 YCbCrPositioning         |     SHORT |        1 |        2 | 2
     130 | 0x8769 ExifTag                  |      LONG |        1 |      360 | 360
     142 | 0xc4a5 PrintImageMatching       | UNDEFINED |      106 |      254 |  ...
   38170 | 0x0103 Compression              |     SHORT |        1 |        6 | 6
   38182 | 0x010e ImageDescription         |     ASCII |       32 |    38330 |

   38194 | 0x010f Make                     |     ASCII |        5 |    38362 | SONY
   38206 | 0x0110 Model                    |     ASCII |        8 |    38368 | ILCE-7R
   38218 | 0x0112 Orientation              |     SHORT |        1 |        1 | 1
   38230 | 0x011a XResolution              |  RATIONAL |        1 |    38376 | 38376/0
   38242 | 0x011b YResolution              |  RATIONAL |        1 |    38384 | 38384/0
   38254 | 0x0128 ResolutionUnit           |     SHORT |        1 |        2 | 2
   38266 | 0x0131 Software                 |     ASCII |       14 |    38392 | ILCE-7R v2.00
   38278 | 0x0132 DateTime                 |     ASCII |       20 |    38406 | 2015:07:09 00:47:56
   38290 | 0x0201 JPEGInterchangeFormat    |      LONG |        1 |    38426 | 38426
   38302 | 0x0202 JPEGInterchangeFormatLeng |      LONG |        1 |    10408 | 10408
   38314 | 0x0213 YCbCrPositioning         |     SHORT |        1 |        2 | 2
$
```

3 Extract the 0x0201 JPEGInterchangeFormat record (of length JPEGInterchangeFormatLength) into buff.jpg and dump that:

```
563 rmills@rmillsmbp:~/temp/foo $ dd bs=1 skip=38426 count=10408 if=buff.tif of=buff.jpg ; exiv2 -pS buff.jpg
10408+0 records in
10408+0 records out
10408 bytes transferred in 0.045672 secs (227886 bytes/sec)
STRUCTURE OF JPEG FILE: buff.jpg
 address | marker     | length  | data
       2 | 0xd8 SOI   |       0
       4 | 0xdb DQT   |     132
     138 | 0xc4 DHT   |     418
     558 | 0xc0 SOF0  |      17
     577 | 0xda SOS   |      12
```

It's a valid little jpg. Open it. It's the preview.

4 Now let's examine the previews:

```
576 rmills@rmillsmbp:~/temp/foo $ exiv2 -pp  DSC01825.jpg
Error: Offset of directory Sony1, entry 0x2001 is out of bounds: Offset = 0x00901076; truncating the entry
Preview 1: image/jpeg, 160x120 pixels, 10408 bytes
577 rmills@rmillsmbp:~/temp/foo $ 571 rmills@rmillsmbp:~/temp/foo $ exiv2 --verbose --force -ep1  DSC01825.jpg

File 1/1: DSC01825.jpg
Error: Offset of directory Sony1, entry 0x2001 is out of bounds: Offset = 0x00901076; truncating the entry
Writing preview 1 (image/jpeg, 160x120 pixels, 10408 bytes) to file ./DSC01825-preview1.jpg
```

5 DSC01825-preview1.jpg and buff.jpg are identical.

```
567 rmills@rmillsmbp:~/temp/foo $ 572 rmills@rmillsmbp:~/temp/foo $ ls -alt *.jpg
-rw-r--r--+ 1 rmills  staff     10408 14 Aug 11:44 DSC01825-preview1.jpg
-rw-r--r--@ 1 rmills  staff     10408 14 Aug 11:31 buff.jpg
-rw-r--r--@ 1 rmills  staff  10125312 13 Aug 17:52 DSC01825.jpg
573 rmills@rmillsmbp:~/temp/foo $ diff buff.jpg DSC01825-preview1.jpg
574 rmills@rmillsmbp:~/temp/foo $ md5 buff.jpg DSC01825-preview1.jpg
MD5 (buff.jpg) = 4d49a9ce3d980b69bfa129e05483b041
MD5 (DSC01825-preview1.jpg) = 4d49a9ce3d980b69bfa129e05483b041
575 rmills@rmillsmbp:~/temp/foo $
```

I'm delighted by this discovery because I've been contemplating buying a Sony Alpha 7 Mirrorless camera and it seems that exiv2 is going to complain about 0x0201 with every image. Not good.

I think it would be better for libexiv2 to respect this situation and suppress the warning. We should however validate the integrity of the situation before deciding to suppress. It's very likely that some software could rewrite the image and blindly copy/relocate the APP1 segment with the resulting 0x0201 address being wrong. Exiv2 should report that situation.

Another reason for being pleased about this discovery is to restore my admiration of Sony. Yesterday I was thinking "Why have Sony not fixed a bug that has been in their Exif firmware for at least 5 years?". Now I'm not so sure it's a bug. The preview is embedded in the APP1 segment.

And the final reason for being pleased is to see how effectively the -pS option can be used to analyse this file. I could almost be motivated to add a -pR option to exiv2 to recursively dump subfiles in the image. Cool stuff, eh?

#7 - 21 Aug 2015 08:05 - Robin Mills

- Category set to not-a-bug

- Status changed from New to Closed

- Assignee set to Robin Mills

- Target version set to 0.26

I'm going to close this as "not a bug" and open a feature request to add the option -pR described above.

#8 - 21 Aug 2015 08:20 - Robin Mills

- % Done changed from 0 to 100

- Estimated time set to 4.00 h

Files

test2.txt	37.1 KB	13 Aug 2015	Daniel Lu
DSC01825.jpg	9.66 MB	13 Aug 2015	Daniel Lu
test3.txt	53.7 KB	13 Aug 2015	Daniel Lu