How to validate CRC-32 calculation of Zip file

Asked 1 year, 10 months ago Modified 1 year ago Viewed 5k times

0

I want to validate that my ZIP file has a correct CRC-32 checksum.
I read that in a ZIP file the CRC-32 data is in bytes 14 to 17:

```
Offset  Bytes   Description[30]
0        4  Local file header signature = 0x04034b50 (read as a little-endian number)
4        2  Version needed to extract (minimum)
6        2  General purpose bit flag
8        2  Compression method
10       2  File last modification time
12       2  File last modification date
14       4  CRC-32 of uncompressed data
18       4  Compressed size
22       4  Uncompressed size
26       2  File name length (n)
28       2  Extra field length (m)
30       n  File name
30+n     m  Extra field
```

I wanted to validate a CRC-32 checksum of a simple ZIP file I created:

```
00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
-----------------------------------------------
50 4B 03 04 14 00 00 00 00 00 38 81 1C 51 4C 18  | PK........8..QL.
C7 8C 02 00 00 00 02 00 00 00 07 00 00 00 31 32  | ÇŒ............12
33 2E 64 61 74 73 73 50 4B 01 02 14 00 14 00 00  | 3.datssPK.......
00 00 00 38 81 1C 51 4C 18 C7 8C 02 00 00 00 02  | ...8..QL.ÇŒ.....
00 00 00 07 00 00 00 00 00 00 00 01 00 20 00 00  | ............. ..
00 00 00 00 00 31 32 33 2E 64 61 74 50 4B 05 06  | .....123.datPK..
00 00 00 00 01 00 01 00 35 00 00 00 27 00 00 00  | ........5...'...
00 00                                            | ..
```

The CRC-32 is: `0x4C18C78C`
I went to this CRC-32 online calculator and added the following un-compressed row from the file:

```
50 4B 03 04 14 00 00 00 00 00 38 81 1C 51
```

This is the result:

```
Algorithm           Result      Check       Poly            Init        RefIn   RefOut
XorOut
CRC-32              0x6A858174  0xCBF43926  0x04C11DB7  0xFFFFFFFF      true    true
0xFFFFFFFF
CRC-32/BZIP2        0xE3FA1205  0xFC891918  0x04C11DB7  0xFFFFFFFF      false   false
0xFFFFFFFF
CRC-32C             0xB578110E  0xE3069283  0x1EDC6F41  0xFFFFFFFF      true    true
0xFFFFFFFF
```

```
CRC-32D             0xAFE2EEA4  0x87315576  0xA833982B  0xFFFFFFFF      true    true
0xFFFFFFFF
CRC-32/MPEG-2       0x1C05EDFA  0x0376E6E7  0x04C11DB7  0xFFFFFFFF      false   false
0x00000000
CRC-32/POSIX        0xFF9B3071  0x765E7680  0x04C11DB7  0x00000000      false   false
0xFFFFFFFF
CRC-32Q             0x79334F11  0x3010BF7F  0x814141AB  0x00000000      false   false
0x00000000
CRC-32/JAMCRC       0x957A7E8B  0x340BC6D9  0x04C11DB7  0xFFFFFFFF      true    true
0x00000000
CRC-32/XFER         0xA7F36A3F  0xBD0BE338  0x000000AF  0x00000000      false   false
0x00000000
```

But none of them equal to: `0x4C18C78C` .

What am I doing wrong? The CRC-32 of the ZIP is the calculation of all the bytes (0-13) before, no?

checksum zip crc crc32

Share Edit Follow

edited Aug 29, 2020 at 16:15

Mark Adler

89.9k 13 106 149

asked Aug 29, 2020 at 13:42

E235

9,062 16 74 124

You could just use `unzip -t whatever.zip` , which checks the CRC of each zip entry in the file.
– Mark Adler Aug 29, 2020 at 16:14

2 Answers

Sorted by:
Trending sort available

Highest score (default)

3

The byte sequence you are running against the online CRC calculator are not uncompressed bytes.

```
50 4B 03 04 14 00 00 00 00 00 38 81 1C 51
```

Those bytes are the first few bytes of the zip file. The CRC32 value in a zip is calculated by running the CRC32 algorithm against the complete uncompressed payload. In your case the payload is the two byte sequence "ss".

To work that out, I converted your hex dump back into a zip file, `tmp.zip` . It contains a single member `123.dat`

```
$ unzip -lv tmp.zip
Archive:  tmp.zip
 Length   Method    Size  Cmpr    Date    Time   CRC-32   Name
```

```
--------  ------  ------- ---- ---------- ----- --------  ----
       2  Stored        2   0% 2020-08-28 16:09 8cc7184c  123.dat
--------          -------  ---                            -------
       2                2   0%                            1 file
```

When I extract that member to stdout & pipe though `hexdump`, we find it contains the two bytes string "ss" (hex 73 73)

```
$ unzip -p tmp.zip | hexdump -C
00000000  73 73                                             |ss|
```

Finally, as already mentioned in another comment, you can check that the CRC value is correct by running `unzip -t`

```
$ unzip -t tmp.zip
Archive:  tmp.zip
    testing: 123.dat                  OK
No errors detected in compressed data of tmp.zip.
```

Share Edit Follow

edited Jun 18, 2021 at 8:55

answered Aug 29, 2020 at 15:57

pmqs
1,578 1 9 9

How you knew that the "ss" is the uncompressed data? – E235 Aug 29, 2020 at 19:45

You provided the zip file (in hex). If you extract the first and only entry, you get "ss". – Mark Adler Aug 29, 2020 at 21:07

1

I was able to create a zip file that matches the one in the question. The header shows that the compression type == 0, which means no compression, the uncompressed size == 2, the data == {73 73}. CRC32 uses reflected input and output, and the CRC is stored in little endian format, so the CRC == 0x8CC7184C.

I get a match using CRC32 on data of {73 73} using this online CRC calculator:

http://www.sunshine2k.de/coding/javascript/crc/crc_js.html

Share Edit Follow

answered Aug 29, 2020 at 15:40

rcgldr
25.7k 3 33 56