Example output:

```
invalid type value detected in Image::printIFDStructure:    25700
Error: Upper boundary of data for directory Image, entry 0x0111 is out of bounds: Offset = 0x00000008, size = 32772, exceeds buffer size by 32350 Bytes; truncating the entry
Error: Directory Image, entry 0x0117 has invalid size 1073741825*4; skipping entry.
Warning: Directory Image, entry 0x0111: Size or data offset value not set, ignoring them.
Error: Offset of directory Image, entry 0x011a is out of bounds: Offset = 0x64000186; truncating the entry
Warning: Directory Image, entry 0x6464 has unknown Exif (TIFF) type 25700; setting type size 1.
Error: Directory Image, entry 0x6464 has invalid size 1684300900*1; skipping entry.
ASAN:SIGSEGV
=================================================================
==8557==ERROR: AddressSanitizer: SEGV on unknown address 0x000000000000 (pc 0x7fee08cf2ce2 sp 0x7ffc6fd6d820 bp 0x7ffc6fd6da70 T0)
==8557==WARNING: Trying to symbolize code, but external symbolizer is not initialized!
    #0 0x7fee08cf2ce1 (/home/lolopop/projects/exiv2-new/clang-debu/src/libexiv2.so.26+0x44cce1)
    #1 0x4a422c (/home/lolopop/projects/exiv2-new/clang-debu/bin/exiv2json+0x4a422c)
    #2 0x49d149 (/home/lolopop/projects/exiv2-new/clang-debu/bin/exiv2json+0x49d149)
    #3 0x7fee07272f44 (/lib/x86_64-linux-gnu/libc.so.6+0x21f44)
    #4 0x491adc (/home/lolopop/projects/exiv2-new/clang-debu/bin/exiv2json+0x491adc)

AddressSanitizer can not provide additional info.
SUMMARY: AddressSanitizer: SEGV ??:0 ??
==8557==ABORTING
```

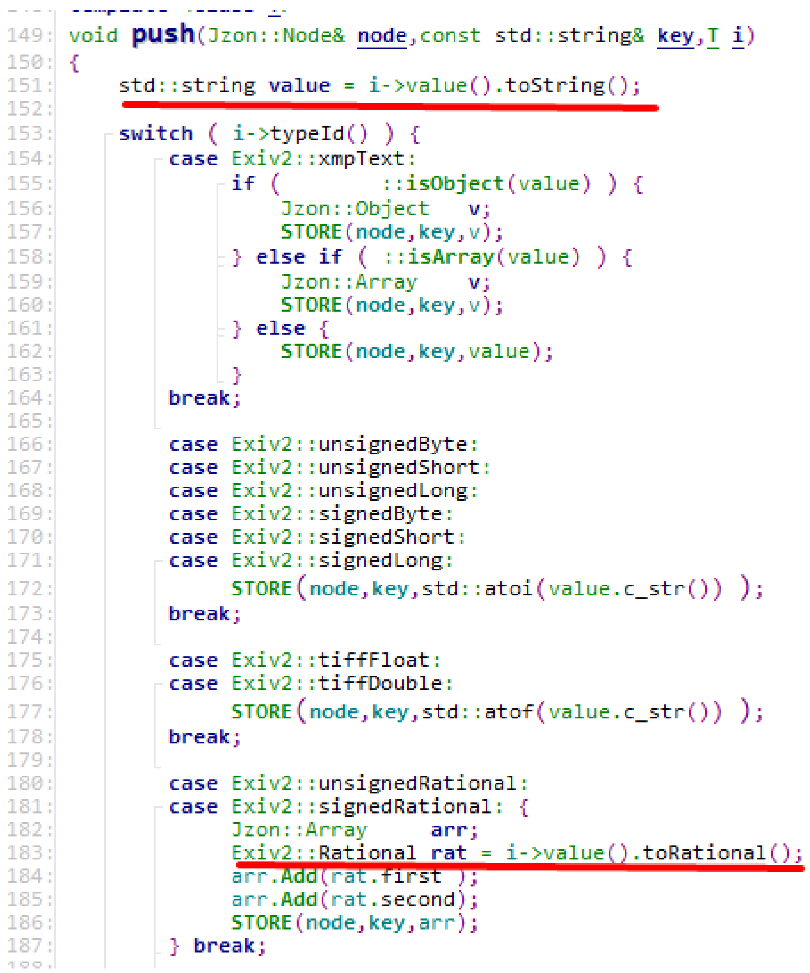
Debug info:

```
Program received signal SIGSEGV, Segmentation fault.
```

```
0x00007f427adc95db in Exiv2::ValueType<std::pair<unsigned int, unsigned int> >::toRational (this=0xa95210, n=0) at /home/lolopop/projects/exiv2-new/gcc-debug/include/exiv2/value.hpp:1719

1719                    return Rational(value_[n].first, value_[n].second);

(rr) bt

#0    0x00007f427adc95db in Exiv2::ValueType<std::pair<unsigned int, unsigned int> >::toRational (this=0xa95210, n=0) at /home/lolopop/projects/exiv2-new/gcc-debug/include/exiv2/value.hpp:1719

#1    0x000000000041e5ea in push<std::_List_const_iterator<Exiv2::Exifdatum> > (node=..., key="XResolution", i=...) at /home/lolopop/projects/exiv2-new/gcc-debug/samples/exiv2json.cpp:183

#2    0x000000000041d313 in main (argc=2, argv=0x7ffff6c66458) at /home/lolopop/projects/exiv2-new/gcc-debug/samples/exiv2json.cpp:292
```

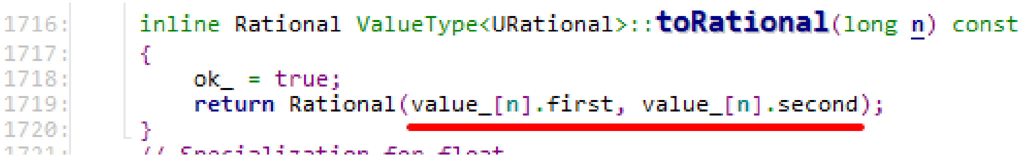
Related source code:

exiv2json.cpp: push function:



```
149: void push(Jzon::Node& node,const std::string& key,T i)
150: {
151:     std::string value = i->value().toString();
152:
153:     switch ( i->typeId() ) {
154:         case Exiv2::xmpText:
155:              if (        ::isObject(value) ) {
156:                  Jzon::Object    v;
157:                  STORE(node,key,v);
158:              } else if ( ::isArray(value) ) {
159:                  Jzon::Array     v;
160:                  STORE(node,key,v);
161:              } else {
162:                  STORE(node,key,value);
163:              }
164:         break;
165:
166:         case Exiv2::unsignedByte:
167:         case Exiv2::unsignedShort:
168:         case Exiv2::unsignedLong:
169:         case Exiv2::signedByte:
170:         case Exiv2::signedShort:
171:         case Exiv2::signedLong:
172:              STORE(node,key,std::atoi(value.c_str()) );
173:         break;
174:
175:         case Exiv2::tiffFloat:
176:         case Exiv2::tiffDouble:
177:              STORE(node,key,std::atof(value.c_str()) );
178:         break;
179:
180:         case Exiv2::unsignedRational:
181:         case Exiv2::signedRational: {
182:              Jzon::Array      arr;
183:              Exiv2::Rational rat = i->value().toRational();
184:              arr.Add(rat.first );
185:              arr.Add(rat.second);
186:              STORE(node,key,arr);
187:         } break;
```

Value.hpp :   toRational method:



```
inline Rational ValueType<URational>::toRational(long n) const
{
    ok_ = true;
    return Rational(value_[n].first, value_[n].second);
}
// Specialization for float
```

Explanation：

The exiv2json program does not check the value of the offset element in the XResolution structure in the tif file. When the value of offset is a random value or error value, value_ vector which save Rational value structure is null. Program access value_ [0] caused Segmentation fault.

The program generates Segmentation fault when parsing the XResolution structure in the tif file.

```
Breakpoint 2, main (argc=2, argv=0x7ffff6c66458) at /home/lolopop/projects/exiv2-new/gcc-debug/samples/exiv2json.cpp:291

291                    Jzon::Node& object = objectForKey(i->key(),root,name);

(rr)

Continuing.


Breakpoint 6, push<std::_List_const_iterator<Exiv2::Exifdatum> > (node=..., key="XResolution", i=...) at /home/lolopop/projects/exiv2-new/gcc-debug/samples/exiv2json.cpp:151

151        std::string value = i->value().toString();

(rr)

Continuing.


Breakpoint 3, push<std::_List_const_iterator<Exiv2::Exifdatum> > (node=..., key="XResolution", i=...) at /home/lolopop/projects/exiv2-new/gcc-debug/samples/exiv2json.cpp:183

183                     Exiv2::Rational rat = i->value().toRational();

(rr)

Continuing.
```

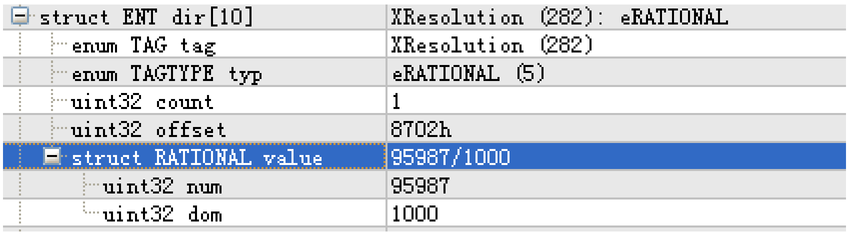
Program received signal SIGSEGV, Segmentation fault.

0x00007f427adc95db in Exiv2::ValueType<std::pair<unsigned int, unsigned int> >::toRational (this=0xa95210, n=0) at /home/lolopop/projects/exiv2-new/gcc-debug/include/exiv2/value.hpp:1719

1719   return Rational(value_[n].first, value_[n].second);

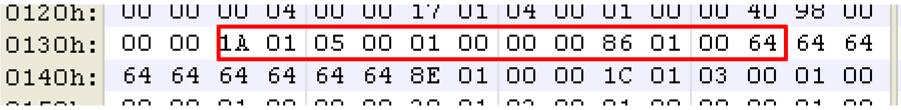
Reference source code, i-> vaule () saves the “Rational value” in the tif file， The position of "rational value" in the tif file is determined by the offseet in the XResolution structure.

The contents of the XResolution structure in the normal tif file are shown in the following picture



⊟ struct ENT dir[10]	XResolution (282): eRATIONAL
enum TAG tag	XResolution (282)
enum TAGTYPE typ	eRATIONAL (5)
uint32 count	1
uint32 offset	8702h
⊟ struct RATIONAL value	95987/1000
uint32 num	95987
uint32 dom	1000

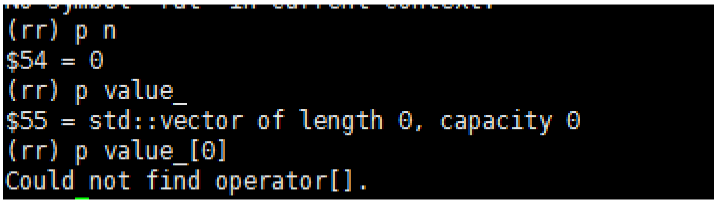
The XResolution structure which cause program to crash is shown in the figure



```
0120h:  00 00 00 04 00 00 17 01 04 00 01 00 00 40 98 00
0130h:  00 00 1A 01 05 00 01 00 00 00 86 01 00 64 64 64
0140h:  64 64 64 64 64 64 8E 01 00 00 1C 01 03 00 01 00
```

The value of offset is 0x64000186. This value is much larger than the size of the tiff file.

The program continues to “toRational” method. “value_” vector which save “Rational value” structure is null. Program access value_ [0] will lead to Segmentation fault.



```
No symbol "rat" in current context.
(rr) p n
$54 = 0
(rr) p value_
$55 = std::vector of length 0, capacity 0
(rr) p value_[0]
Could not find operator[].
```