

Re: memcmp() checker: memory access errors

Source: http://coding.derkeiler.com/Archive/C_CPP/comp.lang.c/2006-12/msg02985.html

- *From:* "kolmogolov@xxxxxxxxxx" <kolmogolov@xxxxxxxxxx>
 - *Date:* 18 Dec 2006 23:30:42 -0800
-

Kohn Emil Dan wrote:

On Mon, 18 Dec 2006, kolmogolov@xxxxxxxxxx wrote:

<snipped>

BTW. I do ungetc() all 256 bytes in data[] back to the stream for decoupling the probe_img() from all the other functions for parsing the details of the headers. I thought tha it were s a proper size for probing many kinds of my image files.

Be careful, because ungetc() can fail to push back the bytes to the input stream. The standard guarantees that only one push back can succeed without an intervening read or file positioning function (in which case your pushed back characters will be be discarded).

So, for safety I'll have to either maintain a 256-byte global buffer or to minimize the size of it whenever possible if I insist on not calling fseek() for the sake of being able to read from a pipe (which is off topic).

I have also installed the memory debugger valgrin mentioned by malc. Thanks too. This is really handy for checking at runtime and it did report my 256-byte memory leak although it did not report any other error. So, I'll be serious about the free() which is left out also in the actual function. And I see that this is no good practice at all. The same to assert() and NDEBUG. I have reread the manpage, removed all asserts, and have gotten the same result from the Checker. It reported 4 memory access errors supposedly at memcmp as Richard has noted. But the Checker does not report the source-line every time. I have not

Re: memcmp() checker: memory access errors

strip'ped
the symbols from the binary though.

Even after I changed

```
if ( 1 != fread(data, probe_len, 1, fp) ) return -2;
```

explicitly to

```
n_items_read = fread(data, probe_len, 1, fp);  
if ( 1 != n_items_read ) return -2;
```

I got the same result of reading 4 uninitialized bytes in the stack
at the line

```
if ( !memcmp(data, KDF_header, 4) )
```

I thought this could have been the data[0] to data[3] because
KDF_header[] has been unconditionally declared and initialized.

What next? Do you think I should send in a bug-report to the author
of my Checker?

.