Bug 29370 - dwarf.c: infinite loop in display_debug_abbrev

Status:	RESOLVED FIXED	**Reported:**	2022-07-15 09:06 UTC by Hex Rabbit
Alias:	None	**Modified:**	2022-07-21 04:20 UTC (History)
		CC List:	3 users (show)
Product:	binutils		
Component:	binutils (show other bugs)	**See Also:**	
Version:	2.40 (HEAD)	**Host:**	
		Target:	
		Build:	
Importance:	P2 normal	**Last reconfirmed:**	2022-07-20 00:00:00
Target Milestone:	2.40		
Assignee:	Alan Modra		
URL:			
Keywords:			
Depends on:			
Blocks:			

Attachments	
file that caused infinite loop (736 bytes, application/octet-stream) 2022-07-15 09:06 UTC, Hex Rabbit	Details
Add an attachment (proposed patch, testcase, etc.)	View All

Note
You need to log in before you can comment on or make changes to this bug.

Hex Rabbit 2022-07-15 09:06:21 UTC **Description**

```
Created attachment 14211 [details]
file that caused infinite loop

During fuzzing campaign, I found some files will cause infinite loop inside
`display_debug_abbrev()` with the command below:

readelf -w poc


build on the latest commit (9afca381e2e46ccee433ce09001506e7683b273f), with default
config `../configure`

Command output:

readelf: Warning: The e_shentsize field in the ELF header is larger than the size
of an ELF section header
readelf: Error: Reading 3584 bytes extends past end of file for program headers
readelf: Error: Reading 1717502016 bytes extends past end of file for .trace_abbrev
section data

... warnings ...
```

```
Contents of the .trace_abbrev section:

... contents ...

Section '.trace_abbrev' has no debugging data.
Contents of the .trace_abbrev section:

  Number TAG (0x0)
   3878592198768      DW_TAG_padding    [no children]
    Unknown AT value: 70e1c3870e1c3870 Unknown FORM value: 45
    DW_AT_virtuality   Unknown FORM value: 46
    DW_AT_location     DW_FORM_addr
    DW_AT_sibling      DW_FORM value: 0
    DW_AT value: 0     DW_FORM value: 0
readelf: Warning: Debug info is corrupted, abbrev offset (1240) is larger than
abbrev section size (8)

Contents of the .trace_abbrev section:

  Number TAG (0x0)
   3878592198768      DW_TAG_padding    [no children]
    Unknown AT value: 70e1c3870e1c3870 Unknown FORM value: 45
    DW_AT_virtuality   Unknown FORM value: 46
    DW_AT_location     DW_FORM_addr
    DW_AT_sibling      DW_FORM value: 0
    DW_AT value: 0     DW_FORM value: 0

... looping same contents ...


I observed through gdb when the program starts to loop, break on `if (list ==
NULL)` line and print out the variables:

gdb$ p start
$43 = (unsigned char *) 0x555555660b70 "\177ELF.trace_abbrev"

gdb$ p section->start
$44 = (unsigned char *) 0x555555660b70 "\177ELF.trace_abbrev"

gdb$ p *list
$45 = {
  first_abbrev = 0x55555565f3d0,
  last_abbrev = 0x55555565f3d0,
  abbrev_base = 0x0,
  abbrev_offset = 0x0,
  next = 0x0,
  start_of_next_abbrevs = 0x555555660b70 "\177ELF.trace_abbrev"
}

The `offset` variable will always be 0, and the `start` variable will never updated
since `list->start_of_next_abbrevs` is the same as `start`, maybe it's caused by
entering this function more than once?
```

cvs-commit@gcc.gnu.org 2022-07-21 04:15:55 UTC **Comment 1**

```
The master branch has been updated by Alan Modra <amodra@sourceware.org>:

https://sourceware.org/git/gitweb.cgi?p=binutils-
gdb.git;h=695c6dfe7e85006b98c8b746f3fd5f913c94ebff

commit 695c6dfe7e85006b98c8b746f3fd5f913c94ebff
Author: Alan Modra <amodra@gmail.com>
Date:   Thu Jul 21 09:56:15 2022 +0930
```

```
PR29370, infinite loop in display_debug_abbrev

The PR29370 testcase is a fuzzed object file with multiple
.trace_abbrev sections.  Multiple .trace_abbrev or .debug_abbrev
sections are not a violation of the DWARF standard.  The DWARF5
standard even gives an example of multiple .debug_abbrev sections
contained in groups.  Caching and lookup of processed abbrevs thus
needs to be done by section and offset rather than base and offset.
(Why base anyway?)  Or, since section contents are kept, by a pointer
into the contents.

        PR 29370
        * dwarf.c (struct abbrev_list): Replace abbrev_base and
        abbrev_offset with raw field.
        (find_abbrev_list_by_abbrev_offset): Delete.
        (find_abbrev_list_by_raw_abbrev): New function.
        (process_abbrev_set): Set list->raw and list->next.
        (find_and_process_abbrev_set): Replace abbrev list lookup with
        new function.  Don't set list abbrev_base, abbrev_offset or next.
```

Note: "PR29370" / "PR 29370" links are printed struck-through (resolved bug links).

Alan Modra **2022-07-21 04:20:58 UTC** Comment 2

```
Fixed for 2.40
```