Packages - Bug #3212

valgrind broken on armv7h

2022-04-01 10:09 PM - GNUtoo

Status:	unconfirmed	**% Done:**	0%
Priority:	bug		
Assignee:			
Category:			

Description

on armv7h, valgrind is broken:

```
$ make check
cc main.c -o main
valgrind --leak-check=full ./main
==10136== Memcheck, a memory error detector
==10136== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==10136== Using Valgrind-3.18.1 and LibVEX; rerun with -h for copyright info
==10136== Command: ./main
==10136==
==10136== Invalid write of size 4
==10136==    at 0x401B100: _dl_start (in /usr/lib/ld-linux-armhf.so.3)
==10136==  Address 0xbd9f7b54 is on thread 1's stack
==10136==  120 bytes below stack pointer
==10136==
==10136== Invalid write of size 4
==10136==    at 0x401236C: _dl_setup_hash (in /usr/lib/ld-linux-armhf.so.3)
==10136==  Address 0xbd9f7b68 is on thread 1's stack
==10136==  8 bytes below stack pointer
==10136==
==10136== Invalid write of size 4
==10136==    at 0x4019548: _dl_sysdep_start (in /usr/lib/ld-linux-armhf.so.3)
==10136==  Address 0xbd9f7aec is on thread 1's stack
==10136==  104 bytes below stack pointer
==10136==
==10136== Invalid write of size 4
==10136==    at 0x4015858: __GI___tunables_init (in /usr/lib/ld-linux-armhf.so.3)
==10136==  Address 0xbd9f7a8c is on thread 1's stack
==10136==  96 bytes below stack pointer
==10136==
==10136== Invalid write of size 4
==10136==    at 0x4012544: _dl_sort_maps_init (in /usr/lib/ld-linux-armhf.so.3)
==10136==  Address 0xbd9f7afc is on thread 1's stack
==10136==  16 bytes below stack pointer
==10136==
==10136== Invalid write of size 4
==10136==    at 0x401FC24: sbrk (in /usr/lib/ld-linux-armhf.so.3)
==10136==  Address 0xbd9f7af0 is on thread 1's stack
==10136==  16 bytes below stack pointer
==10136==
==10136== Invalid write of size 4
==10136==    at 0x401BBF0: dl_main (in /usr/lib/ld-linux-armhf.so.3)
==10136==  Address 0xbd9f78dc is on thread 1's stack
==10136==  528 bytes below stack pointer
==10136==
==10136== Invalid write of size 4
==10136==    at 0x400C488: _dl_new_object (in /usr/lib/ld-linux-armhf.so.3)
==10136==  Address 0xbd9f78ac is on thread 1's stack
==10136==  48 bytes below stack pointer
==10136==
==10136== Invalid write of size 4
==10136==    at 0x400BF50: __minimal_calloc (in /usr/lib/ld-linux-armhf.so.3)
==10136==  Address 0xbd9f78b0 is on thread 1's stack
```

```
==10136==  16 bytes below stack pointer
==10136==
==10136== Invalid write of size 4
==10136==    at 0x400BDF4: __minimal_malloc (in /usr/lib/ld-linux-armhf.so.3)
==10136==  Address 0xbd9f78ac is on thread 1's stack
==10136==  24 bytes below stack pointer
==10136==
==10136== Invalid write of size 4
==10136==    at 0x400C3B8: _dl_add_to_namespace_list (in /usr/lib/ld-linux-armhf.so.3)
==10136==  Address 0xbd9f78e0 is on thread 1's stack
==10136==  16 bytes below stack pointer
==10136==
==10136== Invalid write of size 4
==10136==    at 0x4019CCC: _dl_discover_osversion (in /usr/lib/ld-linux-armhf.so.3)
==10136==  Address 0xbd9f76ec is on thread 1's stack
==10136==  496 bytes below stack pointer
==10136==
==10136== Invalid write of size 4
==10136==    at 0x400716C: _dl_init_paths (in /usr/lib/ld-linux-armhf.so.3)
==10136==  Address 0xbd9f78bc is on thread 1's stack
==10136==  40 bytes below stack pointer
==10136==
==10136== Invalid write of size 4
==10136==    at 0x40181F8: _dl_important_hwcaps (in /usr/lib/ld-linux-armhf.so.3)
==10136==  Address 0xbd9f7844 is on thread 1's stack
==10136==  112 bytes below stack pointer
==10136==
==10136== Invalid write of size 4
==10136==    at 0x4018CAC: _dl_hwcaps_split_masked (in /usr/lib/ld-linux-armhf.so.3)
==10136==  Address 0xbd9f7858 is on thread 1's stack
==10136==  8 bytes below stack pointer
==10136==
==10136== Invalid write of size 4
==10136==    at 0x4018BA0: _dl_hwcaps_split (in /usr/lib/ld-linux-armhf.so.3)
==10136==  Address 0xbd9f7840 is on thread 1's stack
==10136==  16 bytes below stack pointer
==10136==
==10136== Invalid write of size 4
==10136==    at 0x4018140: copy_hwcaps (in /usr/lib/ld-linux-armhf.so.3)
==10136==  Address 0xbd9f780c is on thread 1's stack
==10136==  40 bytes below stack pointer
==10136==
==10136== Invalid write of size 4
==10136==    at 0x401B074: audit_list_add_dynamic_tag (in /usr/lib/ld-linux-armhf.so.3)
==10136==  Address 0xbd9f78f0 is on thread 1's stack
==10136==  8 bytes below stack pointer
==10136==
==10136== Invalid write of size 4
==10136==    at 0x40164D4: _dl_audit_activity_map (in /usr/lib/ld-linux-armhf.so.3)
==10136==  Address 0xbd9f78b0 is on thread 1's stack
==10136==  40 bytes below stack pointer
==10136==
==10136== Invalid write of size 4
==10136==    at 0x401BACC: handle_preload_list (in /usr/lib/ld-linux-armhf.so.3)
==10136==  Address 0xbd9f68dc is not stack'd, malloc'd or (recently) free'd
==10136==
==10136==
==10136== Process terminating with default action of signal 11 (SIGSEGV): dumping core
==10136==  Access not within mapped region at address 0xBD9F68DC
==10136==    at 0x401BACC: handle_preload_list (in /usr/lib/ld-linux-armhf.so.3)
==10136==  If you believe this happened as a result of a stack
==10136==  overflow in your program's main thread (unlikely but
==10136==  possible), you can try to increase the size of the
==10136==  main thread stack using the --main-stacksize= flag.
==10136==  The main thread stack size used in this run was 8388608.
==10136==
==10136== HEAP SUMMARY:
```

```
==10136==     in use at exit: 0 bytes in 0 blocks
==10136==   total heap usage: 0 allocs, 0 frees, 0 bytes allocated
==10136==
==10136== All heap blocks were freed -- no leaks are possible
==10136==
==10136== For lists of detected and suppressed errors, rerun with: -s
==10136== ERROR SUMMARY: 33 errors from 20 contexts (suppressed: 0 from 0)
make: *** [Makefile:4: check] Segmentation fault (core dumped)
```

Here's my valgrind version on armv7h:

```
$ pacman -sS valgrind
extra/valgrind 3.18.1-3 [installed]
    Tool to help find memory-management problems in programs
[...]
```

On i686 I have instead:

```
$ make check
cc main.c -o main
valgrind --leak-check=full ./main
==29363== Memcheck, a memory error detector
==29363== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==29363== Using Valgrind-3.18.1 and LibVEX; rerun with -h for copyright info
==29363== Command: ./main
==29363==
Hello world
==29363==
==29363== HEAP SUMMARY:
==29363==     in use at exit: 0 bytes in 0 blocks
==29363==   total heap usage: 1 allocs, 1 frees, 1,024 bytes allocated
==29363==
==29363== All heap blocks were freed -- no leaks are possible
==29363==
==29363== For lists of detected and suppressed errors, rerun with: -s
==29363== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

With the following valgrind version:

```
$ pacman -sS valgrind
extra/valgrind 3.18.1-4.0 [installed]
    Tool to help find memory-management problems in programs
[...]
```

And on x86_64:

```
# make check
cc main.c -o main
valgrind --leak-check=full ./main
==724655== Memcheck, a memory error detector
==724655== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==724655== Using Valgrind-3.18.1 and LibVEX; rerun with -h for copyright info
==724655== Command: ./main
==724655==
Hello world
==724655==
==724655== HEAP SUMMARY:
==724655==     in use at exit: 0 bytes in 0 blocks
==724655==   total heap usage: 1 allocs, 1 frees, 1,024 bytes allocated
==724655==
==724655== All heap blocks were freed -- no leaks are possible
==724655==
==724655== For lists of detected and suppressed errors, rerun with: -s
```

```
==724655== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

With the following valgrind version:

```
# pacman -sS valgrind
extra/valgrind 3.18.1-4 [installed]
    Tool to help find memory-management problems in programs
[...]
```

I've attached the files used by they are a simple hello world and a Makefile that runs valgrind --leak-check=full ./main

History

#1 - 2022-04-01 10:09 PM - GNUtoo

- File main.c added

Files

Makefile	85 Bytes	2022-04-01	GNUtoo
main.c	73 Bytes	2022-04-01	GNUtoo