Heap buffer overflow in PackLinuxElf32::invert_pt_dynamic #392

Closed giantbranch opened this issue on Jul 23, 2020 · 1 comment

giantbranch commented on Jul 23, 2020 • edited

Author: giantbranch of NSFOCUS Security Team

What's the problem (or question)?

A heap buffer overflow in the latest commit of the devel branch

ASAN reports:

```
==21202==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x62f00000c530 at pc 0x00000058b076 bp 0x7ffdc9ecf670 sp 0x7ffdc9ecf668
READ of size 4 at 0x62f00000c530 thread T0
    #0 0x58b075 in PackLinuxElf32::invert_pt_dynamic(N_Elf::Dyn<N_Elf::ElfITypes<LE16, LE32, LE32, LE32, LE32> > const*) /src/upx-multi/src/p_lx_elf.cpp:1676:25
    #1 0x588959 in PackLinuxElf32::PackLinuxElf32help1(InputFile*) /src/upx-multi/src/p_lx_elf.cpp:305:13
    #2 0x5d5e74 in PackLinuxElf32Le::PackLinuxElf32Le(InputFile*) /src/upx-multi/src/./p_lx_elf.h:395:9
    #3 0x5d5e74 in PackLinuxElf32x86::PackLinuxElf32x86(InputFile*) /src/upx-multi/src/p_lx_elf.cpp:4838:54
    #4 0x5d6261 in PackBSDElf32x86::PackBSDElf32x86(InputFile*) /src/upx-multi/src/p_lx_elf.cpp:4855:50
    #5 0x5d6261 in PackFreeBSDElf32x86::PackFreeBSDElf32x86(InputFile*) /src/upx-multi/src/p_lx_elf.cpp:4866:58
    #6 0x6e4460 in PackMaster::visitAllPackers(Packer* (*)(Packer*, void*), InputFile*, options_t const*, void*) /src/upx-multi/src/packmast.cpp:190:9
    #7 0x6e8ff1 in PackMaster::getUnpacker(InputFile*) /src/upx-multi/src/packmast.cpp:248:18
    #8 0x6e8ff1 in PackMaster::unpack(OutputFile*) /src/upx-multi/src/packmast.cpp:266:9
    #9 0x75826b in do_one_file(char const*, char*) /src/upx-multi/src/work.cpp:160:12
    #10 0x7597c2 in do_files(int, int, char**) /src/upx-multi/src/work.cpp:271:13
    #11 0x555aed in main /src/upx-multi/src/main.cpp:1538:5
    #12 0x7efe5d03d83f in __libc_start_main /build/glibc-e6zv40/glibc-2.23/csu/../csu/libc-start.c:291
    #13 0x41ce98 in _start (/out/upx-multi/upx-multi+0x41ce98)

0x62f00000c532 is located 0 bytes to the right of 49458-byte region [0x62f000000400,0x62f00000c532)
allocated by thread T0 here:
    #0 0x49519d in malloc (/out/upx-multi/upx-multi+0x49519d)
    #1 0x569797 in MemBuffer::alloc(unsigned long long) /src/upx-multi/src/mem.cpp:194:42

SUMMARY: AddressSanitizer: heap-buffer-overflow /src/upx-multi/src/p_lx_elf.cpp:1676:25 in PackLinuxElf32::invert_pt_dynamic(N_Elf::Dyn<N_Elf::ElfITypes<LE16, LE32, LE32, LE32,
LE32> > const*)
Shadow bytes around the buggy address:
  0x0c5e7fff9850: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c5e7fff9860: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c5e7fff9870: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c5e7fff9880: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x0c5e7fff9890: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
=>0x0c5e7fff98a0: 00 00 00 00 00 00[02]fa fa fa fa fa fa fa fa
  0x0c5e7fff98b0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c5e7fff98c0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c5e7fff98d0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c5e7fff98e0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
  0x0c5e7fff98f0: fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa fa
Shadow byte legend (one shadow byte represents 8 application bytes):
  Addressable:           00
  Partially addressable: 01 02 03 04 05 06 07
  Heap left redzone:       fa
  Freed heap region:       fd
  Stack left redzone:      f1
  Stack mid redzone:       f2
  Stack right redzone:     f3
  Stack after return:      f5
  Stack use after scope:   f8
  Global redzone:          f9
  Global init order:       f6
  Poisoned by user:        f7
  Container overflow:      fc
  Array cookie:            ac
  Intra object redzone:    bb
  ASan internal:           fe
  Left alloca redzone:     ca
  Right alloca redzone:    cb
  Shadow gap:              cc
==21202==ABORTING
```

What should have happened?

Check if the file is normal, exit if abnormal

Do you have an idea for a solution?

Add more checks

How can we reproduce the issue?

upx.out -d <poc_filename>

poc:
tests_07edd5f520df09443f3622129449d21c6b7c3c7c_.tar.gz

Please tell us details about your environment.

- UPX version used (`upx --version`):

```
upx 4.0.0-git-87b73e5cfdc1+
UCL data compression library 1.03
zlib data compression library 1.2.8
LZMA SDK version 4.43
Copyright (C) 1996-2020 Markus Franz Xaver Johannes Oberhumer
Copyright (C) 1996-2020 Laszlo Molnar
Copyright (C) 2000-2020 John F. Reiser
Copyright (C) 2002-2020 Jens Medoch
Copyright (C) 1995-2005 Jean-loup Gailly and Mark Adler
Copyright (C) 1999-2006 Igor Pavlov
UPX comes with ABSOLUTELY NO WARRANTY; for details t
```

- Host Operating System and version: Ubuntu 16.04.2 LTS
- Host CPU architecture: x86_64
- Target Operating System and version: same as Host
- Target CPU architecture: same as Host

jreiser commented on Jul 23, 2020 Collaborator

Fixes for #390 and #391 handle this one, too.

giantbranch closed this as completed on Jul 27, 2020

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

2 participants