New issue

Jump to bottom

libsolv “pool_installable_whatprovides” function a heap-overflow vulnerability #417

Closed **yangjiageng** opened this issue on Dec 13, 2020 · 7 comments

yangjiageng commented on Dec 13, 2020 • edited

“pool_disabled_solvable” function a heap-overflow vulnerability
“pool_installable” function a heap-overflow vulnerability
“pool_installable_whatprovides” function a heap-overflow vulnerability
Description:
There are three heap-buffer overflow bugs in function:
static inline int pool_disabled_solvable(const Pool *pool, Solvable *s)
static inline int pool_installable(const Pool *pool, Solvable *s)
static inline int pool_installable_whatprovides(const Pool *pool, Solvable *s)
at src/repo.h: line 96, line 120 and line 138
The statement of these three lines are same, as follows:
if (!MAPTST(pool->considered, id))

The program defines “MAPTST(m, n)” that “((m)->map[(n) >> 3] & (1 << ((n) & 7)))”.

MAPTST(pool->considered, id) is same as pool->considered->map[id>>3] & (1 << (id & 7)).
This statement involves pool->considered->map[id>>3].

The variable pool->considered is a Map structure pointer.
The Map structure as following:
typedef struct s_Map {
unsigned char *map;
int size;
} Map;
If the index value “id>>3” is bigger than pool->considered->size, there is a heap overflow bug.

Please reproduce this issue through the following PoC: /libsolvBuildDir/tools/testsolv PoC-pool_disabled_solvable-line96
If you configure CC with flag -fsanitize=address, you will get the following outputs:

```
testcase_read: cannot parse command 'D'
disable: unknown package 'E-1-1.src@available'
disable: unknown package 'F-1-1.src@available'
test 1:
test 2:
str2job: unknown selection flag 'b'
testcase_read: cannot parse command 'repo'
test 3:
Results differ:
-job noop name (A . i686) > 1 [setarch]
=================================================================
AddressSanitizer: heap-buffer-overflow on address 0x6020000000f1 at pc 0x7f0ef3b850cc bp 0x7fffcdcb9470 sp 0x7fffcdcb9468
READ of size 1 at 0x6020000000f1 thread T0
#0 0x7f0ef3b850cb in pool_disabled_solvable /root/Experiments/real-world/libsolv/src/repo.h:96:12
#1 0x7f0ef3b850cb in solvable_matches_selection_flags /root/Experiments/real-world/libsolv/src/selection.c:744:46
#2 0x7f0ef3b850cb in selection_name /root/Experiments/real-world/libsolv/src/selection.c:792:12
#3 0x7f0ef3b49618 in selection_name_arch /root/Experiments/real-world/libsolv/src/selection.c:835:11
#4 0x7f0ef3b49618 in selection_name_arch_rel /root/Experiments/real-world/libsolv/src/selection.c:922:13
#5 0x7f0ef3b49618 in selection_make /root/Experiments/real-world/libsolv/src/selection.c:1384:11
#6 0x7f0efce9796e in addselectionjob /root/Experiments/real-world/libsolv/ext/testcase.c:896:9
#7 0x7f0efce90cc6 in testcase_read /root/Experiments/real-world/libsolv/ext/testcase.c:2186:8
#8 0x4f144b in main /root/Experiments/real-world/libsolv/tools/testsolv.c:159:11
#9 0x7f0ef28f5bf6 in __libc_start_main /build/glibc-S7xCS9/glibc-2.27/csu/../csu/libc-start.c:310
#10 0x41e6f9 in _start (/root/Experiments/real-world/libsolv/build/tools/testsolv+0x41e6f9)

0x6020000000f1 is located 0 bytes to the right of 1-byte region [0x6020000000f0,0x6020000000f1)
allocated by thread T0 here:
#0 0x4abe48 in calloc /root/Downloads/llvm-build/llvm/projects/compiler-rt/lib/asan/asan_malloc_linux.cpp:154
#1 0x7f0ef3a52f10 in solv_calloc /root/Experiments/real-world/libsolv/src/util.c:79:9
#2 0x7f0ef38ccdba in map_init /root/Experiments/real-world/libsolv/src/bitmap.c:24:22
#3 0x7f0efce9304a in testcase_read /root/Experiments/real-world/libsolv/ext/testcase.c:2318:8
#4 0x4f144b in main /root/Experiments/real-world/libsolv/tools/testsolv.c:159:11
#5 0x7f0ef28f5bf6 in __libc_start_main /build/glibc-S7xCS9/glibc-2.27/csu/../csu/libc-start.c:310
```

SUMMARY: AddressSanitizer: heap-buffer-overflow /root/Experiments/real-world/libsolv/src/repo.h:96:12 in pool_disabled_solvable
Shadow bytes around the buggy address:
0x0c047fff7fc0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0c047fff7fd0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0c047fff7fe0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0c047fff7ff0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0c047fff8000: fa fa fd fd fa fa 00 02 fa fa 00 00 fa fa 04 fa
=>0x0c047fff8010: fa fa 04 fa fa fa fd fd fa fa 00 00 fa fa[01]fa
0x0c047fff8020: fa fa fd fd fa fa fd fa fa fa fd fa fa fa fd fa
0x0c047fff8030: fa fa fd fa fa fa fd fd fa fa fd fd fa fa fd fd
0x0c047fff8040: fa fa fd fd fa fa fd fa fa fa fd fa fa fa fd fa
0x0c047fff8050: fa fa fd fa fa fa fd fa fa fa fd fd fa fa 00 02
0x0c047fff8060: fa fa 00 00 fa fa 04 fa fa fa 04 fa fa fa fd fd
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
==117371==ABORTING

mlschroe commented on Dec 14, 2020 Member

Made testcase reader more robust.

mlschroe closed this as completed on Dec 14, 2020

Barry-chen12581 commented on Sep 13, 2021

It seems this issue was assigned CVE-2021-33928.

Looking at the commit history, it appears `0077ef2` (included in 0.7.17) is the commit that addresses this issue?

Barry-chen12581 commented on Sep 14, 2021

yes so is the commit `0077ef2` fix them? ---------- 原始邮件 ----------
@.***>
发送时间: 2021年9月14日(星期二) 凌晨0:44
@.***>;
@.******@***.***>;
主题: Re: [openSUSE/libsolv] libsolv “pool_installable_whatprovides” function a heap-overflow vulnerability (#417)

mlschroe commented on Sep 15, 2021 Member

Yes, that's the correct commit.

mengtanhzc commented on Nov 2, 2021 • edited

> Yes, that's the correct commit.

@mlschroe
Hi, I'm sorry to bother you, but I have some questions for consultation.

I found that cve-2021-33928, CVE-2021-33929, CVE-2021-33930, CVE-2021-33938 was linked with this issue.
It looks like they are all missing a bounds check before using MAPTST, but testcase_read: error out if ... only modifies ext/testcase.c, which makes me a bit confused.
Since I don't know enough about libsolv, I have the following questions:

1. why testcase_read: error out if ... can be used to fix those cves?
2. if I am not using testsolv, is there a possibility of triggering heap-buffer overflow in current version?
   I wonder if you could help me with this? Thanks a lot!

mlschroe commented on Nov 3, 2021 Member

The CVEs are pretty confused. There is no need for a bounds check if the hashes are setup correctly. The underlying problem was in the testcase reader, which allowed to add new packages after the hashes were set up. This is fixed, so it's not possible to run into this with the current version.

mengtanhzc commented on Nov 3, 2021

> The CVEs are pretty confused. There is no need for a bounds check if the hashes are setup correctly. The underlying problem was in the testcase reader, which allowed to add new packages after the hashes were set up. This is fixed, so it's not possible to run into this with the current version.

OK, that's greate. Many thanks.

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

4 participants