Bug 1177201 - (CVE-2020-28049) VUL-0: CVE-2020-28049: sddm: race condition in setting up Xorg -auth file in conjunction with Xorg -displayfd parameter

Status: NEW

Classification: openSUSE
Product: openSUSE Distribution
Component: Security
Version: Leap 15.1
Hardware: Other Other

Priority: P3 - Medium **Severity**: Normal (vote)
Target Milestone: Leap 15.1
Assigned To: Security Team bot
QA Contact: E-mail List

URL:
Whiteboard:
Keywords:

Depends on:
Blocks:
Show dependency tree / graph

- Create test case
- Clone This Bug

Reported: 2020-10-01 14:42 UTC by Fabian Vogt
Modified: 2020-11-11 14:46 UTC (History)
CC List: 3 users (show)

See Also:
Found By: ---
Services Priority:
Business Priority:
Blocker: ---

Attachments

Add an attachment (proposed patch, testcase, etc.)

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

Matthias Gerstner 2020-10-02 12:12:06 UTC Comment 4

I was able to reproduce this with sddm. As an addendum to the reproducer from comment 0: you need to set an appropriate DISPLAY environment variable for this work from e.g. an SSH connection outside of a graphical session. For sddm `export DISPLAY=:0` is enough.

I also managed to start an xterm in the sddm greeter context, but sddm seems to kill or unmap that window quickly again. Using an X aware program tailored towards this attack (I tested using an adjusted version of my xwmfs [1] project), it is possible to keep the existing connection to the X server even after a regular user performs a graphical login. Consequences are for example, that an arbitrary unprivileged local user can kill other user's X applications, retrieve/change window names, or read/write the clipboard/primary selection. In other words a big information leak and integrity violation.

I don't know if this attack vector is enough to gain root privileges. This would require that the greeter is running as root and that the access to the X display while the greeter runs somehow makes root privileges possible.

The good thing is that the issue can only be exploited once during boot or when the X server is restarted.

Strangely enough I was not able to reproduce the same using `gdm`. Either the race condition is way shorter there, or it uses some other measures to prevent the race in the first place. It is passing the `-auth` and `-displayfd` switches just like `sddm` does, however.

My first guess would be, without having looked closely into the involved code, that this special usage of `-auth` and `-displayfd` needs a secure handover protocol i.e. connections to the X display should only become possible once the greeter has fully written the auth file and somehow signaled this to the Xserver. An hacky way to fix it would also be for the Xserver to reject all connections when `-auth` was passed but the specified file is not existing or not containing valid information.

What's your take on this issue Stefan?

[1]: https://github.com/gerstner-hub/xwmfs

Fabian Vogt 2020-10-02 13:12:56 UTC Comment 5

(In reply to Matthias Gerstner from comment #4)

> I was able to reproduce this with sddm. As an addendum to the reproducer from
> comment 0: you need to set an appropriate DISPLAY environment variable for
> this work from e.g. an SSH connection outside of a graphical session. For
> sddm
> `export DISPLAY=:0` is enough.
>
> I also managed to start an xterm in the sddm greeter context, but sddm seems
> to kill or unmap that window quickly again.

AFAICT the greeter appears in foreground, and without a running WM it's not possible to interact with anything behind the greeter.

> Using an X aware program tailored
> towards this attack (I tested using an adjusted version of my xwmfs [1]
> project), it is possible to keep the existing connection to the X server even
> after a regular user performs a graphical login. Consequences are for
> example,
> that an arbitrary unprivileged local user can kill other user's X
> applications, retrieve/change window names, or read/write the

> clipboard/primary selection. In other words a big information leak and
> integrity violation.

And also capture all input and output, including username/password in the login screen...

> I don't know if this attack vector is enough to gain root privileges. This
> would require that the greeter is running as root and that the access to the
> X
> display while the greeter runs somehow makes root privileges possible.

The greeter runs as "sddm" user and doesn't have any special privileges itself. So it shouldn't immediately result in a privesc.

> The good thing is that the issue can only be exploited once during boot or
> when the X server is restarted.

The requirement is that there is normal user access possible, so triggering X to start in such cases is not too far off (logout when owning a graphical session, triggering creation of a new graphical session on a different VT over dbus)

> Strangely enough I was not able to reproduce the same using `gdm`. Either the
> race condition is way shorter there, or it uses some other measures to
> prevent
> the race in the first place. It is passing the `-auth` and `-displayfd`
> switches just like `sddm` does, however.

See the last paragraph of my initial comment.

> My first guess would be, without having looked closely into the involved
> code,
> that this special usage of `-auth` and `-displayfd` needs a secure handover
> protocol i.e. connections to the X display should only become possible once
> the greeter has fully written the auth file and somehow signaled this to the
> Xserver. An hacky way to fix it would also be for the Xserver to reject all
> connections when `-auth` was passed but the specified file is not existing or
> not containing valid information.
>
> What's your take on this issue Stefan?
>
> [1]: https://github.com/gerstner-hub/xwmfs

Stefan Dirsch 2020-10-02 14:08:02 UTC Comment 6

Not sure what I can contribute to this discussion. You digged much deeper into this as I did. Just one thing to add. The way gdm starts the Xserver - as I recently learned.

The X arguments in gdm:

```
/usr/bin/X vt2 (could be vt2-7)
-displayfd x
-auth xxxx
-background none
-noreset
-keeptty
-verbose 7 (could be 7 or 3)
-core
-listen tcp (if allow_remote_connections set to "yes" in /etc/gdm/custom.conf)
-nolisten tcp (default)
```

Matthias Gerstner 2020-10-05 09:41:20 UTC Comment 7

(In reply to fvogt@suse.com from comment #5)

> AFAICT the greeter appears in foreground, and without a running WM it's not
> possible to interact with anything behind the greeter.

Makes sense.

> The requirement is that there is normal user access possible, so triggering X
> to start in such cases is not too far off (logout when owning a graphical session
> triggering creation of a new graphical session on a different VT over dbus)

Yes this would be true for multi users graphical systems. Which are rather rare these days. I was thinking more of something like a compromised `nobody` account or something similar.

> See the last paragraph of my initial comment.

Ah I see. So the point here is that gdm probably writes out the XAuthority file before starting Xorg.

So what is left to do for this issue on our side is about the following:

- identifying other common display managers that are affected by default
- determining if this more of an Xorg issue or more of a display manager
  issue. In the end we can send out reports to all affected parties.

I will try to find some more time today to do this.

Matthias Gerstner 2020-10-05 12:55:26 UTC Comment 8

I checked and none of the other major display managers seem to be affected by this, at least not by default (checked gdm, xdm, lightdm, lxdm).

In the xserver code function `CheckAuthorization` we can get some hints about how Xorg handles changes to Xauthority files:

```
        /*
         * If the authorization file has at least one entry for this server,
         * disable local access. (loadauth > 0)
         *
         * If there are zero entries (either initially or when the
         * authorization file is later reloaded), or if a valid
         * authorization file was never loaded, enable local access.
```

```
         * (loadauth == 0 || !loaded)
         *
         * If the authorization file was loaded initially (with valid
         * entries for this server), and reloading it later fails, don't
         * change anything. (loadauth == -1 && loaded)
         */
```

`CheckAuthorization()` seems to be executed every time a new user tries to connect to the XServer. During every execution the server checks whether the modification time of the file changed and if so, reloads its contents.

This means that the Xserver has *some* support to recognize updates to the authority file. But since there is no means of synchronization that would allow the Xserver to read changed Xauthority files in a race-free fashion this feature is shaky. If the Xserver would *not* do this, however, then the issue discovered in this bug could become permanent.

The Xserver(1) man page states regarding `-auth`:

```
       -auth authorization-file
              specifies a file which contains a collection of authorization
records used  to  authenticate  ac-
              cess. See also the xdm(1) and Xsecurity(7) manual pages.
```

I would say that this doesn't cover the file not yet existing or not yet having the correct content when the Xserver is started up. Therefore I'm starting to think that the issue is more on the sddm side than on the Xserver side.

My suggestion is to report this privately to sddm upstream and maybe give the Xorg people a hint about this issue and that a safer default would be better when `-auth` is specified but the file does not exist yet or is empty.

Fabian Vogt 2020-10-05 13:26:58 UTC Comment 9

(In reply to Matthias Gerstner from comment #8)

> I checked and none of the other major display managers seem to be affected by
> this, at least not by default (checked gdm, xdm, lightdm, lxdm).
>
> In the xserver code function `CheckAuthorization` we can get some hints about
> how Xorg handles changes to Xauthority files:
>
> ```
>         /*
>          * If the authorization file has at least one entry for this server,
>          * disable local access. (loadauth > 0)
>          *
>          * If there are zero entries (either initially or when the
>          * authorization file is later reloaded), or if a valid
>          * authorization file was never loaded, enable local access.
>          * (loadauth == 0 || !loaded)
>          *
>          * If the authorization file was loaded initially (with valid
>          * entries for this server), and reloading it later fails, don't
>          * change anything. (loadauth == -1 && loaded)
>          */
> ```
>
> `CheckAuthorization()` seems to be executed every time a new user tries to
> connect to the XServer. During every execution the server checks whether the
> modification time of the file changed and if so, reloads its contents.
>
> This means that the Xserver has *some* support to recognize updates to the
> authority file. But since there is no means of synchronization that would
> allow the Xserver to read changed Xauthority files in a race-free fashion
> this
> feature is shaky.

libXau provides XauLockAuth/XauUnlockAuth, which are probably supposed to take care of that.

> If the Xserver would *not* do this, however, then the issue
> discovered in this bug could become permanent.
>
> The Xserver(1) man page states regarding `-auth`:
>
> ```
>        -auth authorization-file
>               specifies a file which contains a collection of authorization
> records used  to  authenticate  ac-
>               cess. See also the xdm(1) and Xsecurity(7) manual pages.
> ```
>
> I would say that this doesn't cover the file not yet existing or not yet
> having the correct content when the Xserver is started up. Therefore I'm
> starting to think that the issue is more on the sddm side than on the Xserver
> side.

Yes, that's how far I got as well before I filed the report. It's just that there is absolutely no document explaining how this is designed to work and what DMs are supposed to do here, so without that there's just no way to write an entry with the proper display number for the auth file. I can't tell whether that's intentional or not and so the question about what a DM is actually expected to write there remains. I really don't want to base critical code like that on guesses and assumptions.

> My suggestion is to report this privately to sddm upstream and maybe give the
> Xorg people a hint about this issue and that a safer default would be better
> when `-auth` is specified but the file does not exist yet or is empty.

I did the former as a heads-up and prepared a patch which does something similar like gdm, but uses "0" instead of passing an empty display number as that is at least shown properly in "xauth list". AFAICT currently the X server doesn't care about the display number or the host name and matches with every MIT-MAGIC-COOKIE-1 entry.

@sndirsch: Can you ask upstream X about this? Some official document on how DMs are supposed to write -auth + -displayfd would be useful and ideally also hardening,
like refusing connections if the passed file is empty or does not exist.

Matthias Gerstner 2020-10-05 13:59:37 UTC Comment 10

(In reply to fvogt@suse.com from comment #9)

> libXau provides XauLockAuth/XauUnlockAuth, which are probably supposed to take
> care of that.

As long as the xserver doesn't call those functions that's not helping much
...

If the xserver would use those locking functions then the useage of -auth and
-displayfd could be made safe. sddm would need to lock the file before
starting the xserver, then wait for the display number, write the file, only
then unlock.

> Yes, that's how far I got as well before I filed the report. It's just that there
> is absolutely no document explaining how this is designed to work and what DMs
> are supposed to do here, so without that there's just no way to write an entry
> with the proper display number for the auth file.

Since sddm is the only DM that is affected by this there's a chance that it's
not supposed to work this specific way at all. But you simply will have to get
the info from Xorg people, this is beyond the security topic as such.

> AFAICT currently the X server doesn't care about the display number or the
> host name and matches with every MIT-MAGIC-COOKIE-1 entry.

The display number would only be of interest to the Xserver if the same
Xauthority file would be used for multiple displays at the same time. Since
each X server instance can only serve one display this wouldn't make much
sense.

Therefore writing the display number into the authority file is probably
unnecessary. But again you would need to get confirmation from Xorg people.
This X11 stuff is pretty strange at times.

Stefan Dirsch 2020-10-05 14:33:41 UTC Comment 11

Well, I doubt there is any documentation/special knowledge in the X community how
to (best) use these -auth/-displayfd Xserver options for displaymanagers. I would
guess displaymanagers, which came after xdm initially copied the usage from xdm
(kdm and gdm) and when there were changes in xdm due to this it was adjusted in
those as well (if it didn't go unnoticed there!).

Nevertheless you could ask on the xorg-devel mailing list or in #xorg-devel on
freenode, if you like. I'm not so happy playing proxy here since usually this ends
up in questions to me, which you could ask better than me.

https://lists.x.org/mailman/listinfo/xorg-devel
#xorg-devel (freenode)

The security team must have some contact to X developers usually caring about the
security updates (unfortunately I don't have these). That would be another option
to use as contact.

Matthias Gerstner 2020-10-06 09:40:54 UTC Comment 12

(In reply to sndirsch@suse.com from comment #11)

> Well, I doubt there is any documentation/special knowledge in the X
> community how to (best) use these -auth/-displayfd Xserver options for
> displaymanagers.

This is most probably simply a grey area, like so many in the X11 context.
Maybe a scenario like this was considered somewhen in the past but never quite
implemented fully (e.g. missing locking calls).

> The security team must have some contact to X developers usually caring
> about the security updates (unfortunately I don't have these). That would be
> another option to use as contact.

We are members of the list xorg-security@lists.x.org. You can post your
question there and discuss it. But make it clear that the issue is under
EMBARGO. I guess it is more up to sddm upstream to decide when to publish
this.

Matthias Gerstner 2020-10-07 11:49:50 UTC Comment 13

@fvogt: do you need any additinal support with this? Did you start a security
incident process with sddm upstream?

Fabian Vogt 2020-10-07 12:29:04 UTC Comment 14

(In reply to Matthias Gerstner from comment #13)

> @fvogt: do you need any additinal support with this? Did you start a security
> incident process with sddm upstream?

Not yet, I'd do that once I've got the patch tested properly. If this needs a CVE
(probably), could you allocate one (like for ~~bug 1191455~~)?

Matthias Gerstner 2020-10-07 12:54:00 UTC Comment 15

(In reply to fvogt@suse.com from comment #14)

> Not yet, I'd do that once I've got the patch tested properly. If this needs a CVE

We can request one via the Mitre CVE form. But that means we need to share it
with Mitre, which we would like to do only when we are close to publishing the
issue.

Is sddm part of some bigger project like KDE or close to Red Hat? Then they
should be able to use one of their own CVEs. Generally you should always ask
upstream first how they like to handle CVE assignment.

Stefan Dirsch 2020-10-19 02:50:27 UTC Comment 16

Looks like this needs to be addressed in sddm. In case my assumption turns out to be wrong and it needs to be handled i n Xserver instead please feel free to put me in the loop again. Thanks!

Matthias Gerstner 2020-10-21 08:01:59 UTC Comment 17

Are there any news regarding the upstream embargo process? If there are any emails exchanged regarding this it would be good to include security@suse.de to keep us in the loop.

We will apply our usual 90 days max CRD for the time being until we know more from upstream:

Internal CRD: 2020-12-30 or earlier

Fabian Vogt 2020-10-21 08:12:16 UTC Comment 18

I wrote a mail to the maintainers with patches on the 10th, but didn't get a reaction so far. I'll send a ping end of this week, with security@suse.de in CC.

Matthias Gerstner 2020-11-02 12:22:51 UTC Comment 21

(In reply to fvogt@suse.com from comment #20)

> I asked about the CVE assignment and got this back:
>
> >> For the release, having the CVE ID in the changelog already would be ideal.
> >> Did you allocate one already or should I ask the SUSE sec team to get one?
> >>
> >Please have the SUSE sec team get one, thanks.

Actually we don't have any special powers in this regard. We can only actively assign CVEs for SUSE's own software, not for external projects. For external projects the following CVE form needs to be used:

https://cveform.mitre.org

You could basically do this yourself. It will usually take one work day until the CVE is communicated by Mitre.

Did you come to an agreement with upstream and/or the X.org guys about whether this is an sddm-only issue or a larger X.org issue? This would be an important pointer for the CVE contents, too.

Fabian Vogt 2020-11-02 12:57:35 UTC Comment 22

(In reply to Matthias Gerstner from comment #21)

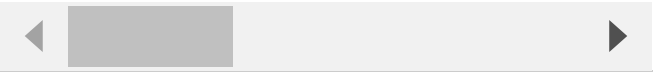
> (In reply to fvogt@suse.com from comment #20)
> > I asked about the CVE assignment and got this back:
> >
> > >> For the release, having the CVE ID in the changelog already would be ideal.
> > >> Did you allocate one already or should I ask the SUSE sec team to get one?
> > >>
> > >Please have the SUSE sec team get one, thanks.
> >
> Actually we don't have any special powers in this regard. We can only
> actively
> assign CVEs for SUSE's own software, not for external projects. For external
> projects the following CVE form needs to be used:
>
> https://cveform.mitre.org
>
> You could basically do this yourself. It will usually take one work day until
> the CVE is communicated by Mitre.

Ah, ok. It's not entirely clear to me how the process works, so I'll leave that to someone else, at least this time.

> Did you come to an agreement with upstream and/or the X.org guys about
> whether
> this is an sddm-only issue or a larger X.org issue? This would be an
> important
> pointer for the CVE contents, too.

CVE-2015-3164 seems similar enough that I assume that the developers are aware of it and they mitigated it for Xwayland. It's also somewhat documented in the Xsecurity(7) man page, though that doesn't seem to be installed here (?): https://www.x.org/releases/current/doc/man/man1/Xserver.1.xhtml#heading10

> If this file contains any authorization records, the local host is not automatica



Matthias Gerstner 2020-11-02 13:43:28 UTC Comment 23

(In reply to fvogt@suse.com from comment #22)

> > You could basically do this yourself. It will usually take one work day until
> > the CVE is communicated by Mitre.
>
> Ah, ok. It's not entirely clear to me how the process works, so I'll leave that t

Okay I will fill out the request form.

Matthias Gerstner 2020-11-03 08:02:24 UTC Comment 24

Mitre assigned CVE-2020-28049 for this issue.

Fabian Vogt 2020-11-03 10:05:31 UTC Comment 25

(In reply to Matthias Gerstner from comment #24)

> Mitre assigned CVE-2020-28049 for this issue.

SDDM v0.19.0 is out, with the CVE mentioned in the changelog:
https://github.com/sddm/sddm/releases/tag/v0.19.0

Matthias Gerstner 2020-11-03 11:08:37 UTC Comment 26

Okay, publishing this bug, too.

This will needs updates in Leap 15.0 and 15.1. Will you take care of it?

Do you want to write a posting to the oss-sec mailing list making people aware
of this issue? Otherwise I'd do it.

Fabian Vogt 2020-11-03 12:25:12 UTC Comment 27

(In reply to Matthias Gerstner from comment #26)

> Okay, publishing this bug, too.
>
> This will needs updates in Leap 15.0 and 15.1. Will you take care of it?

Yep: https://build.opensuse.org/request/show/845707
For TW it's https://build.opensuse.org/request/show/845708

> Do you want to write a posting to the oss-sec mailing list making people
> aware
> of this issue? Otherwise I'd do it.

I asked upstream what they prefer, but didn't get an answer yet. IMO you can go
ahead with the mail as that shouldn't be needlessly delayed.

Matthias Gerstner 2020-11-04 09:25:51 UTC Comment 30

I'm adjust the bug summary to make it clear that this is an issue in sddm.

Matthias Gerstner 2020-11-04 10:56:04 UTC Comment 31

I published this on oss-sec [1].

[1]: https://www.openwall.com/lists/oss-security/2020/11/04/2

Swamp Workflow Management 2020-11-07 17:14:44 UTC Comment 33

openSUSE-SU-2020:1870-1: An update that fixes one vulnerability is now available.

Category: security (moderate)
Bug References: 1177201
CVE References: CVE-2020-28049
JIRA References:
Sources used:
openSUSE Leap 15.2 (src):   sddm-0.18.0-lp152.5.3.1
openSUSE Leap 15.1 (src):   sddm-0.18.0-lp151.3.6.1

Swamp Workflow Management 2020-11-10 17:17:21 UTC Comment 34

openSUSE-SU-2020:1897-1: An update that fixes one vulnerability is now available.

Category: security (moderate)
Bug References: 1177201
CVE References: CVE-2020-28049
JIRA References:
Sources used:
openSUSE Backports SLE-15-SP1 (src):   sddm-0.18.0-bp151.4.6.1

Swamp Workflow Management 2020-11-11 14:46:39 UTC Comment 35

openSUSE-SU-2020:1899-1: An update that fixes one vulnerability is now available.

Category: security (moderate)
Bug References: 1177201
CVE References: CVE-2020-28049
JIRA References:
Sources used:
openSUSE Backports SLE-15-SP2 (src):   sddm-0.18.0-bp152.5.3.1
