plasma-login-manager: Weaknesses in plasmaloginauthhelper (CVE-2026-25710)

Apr 27, 2026 • Matthias Gerstner, Filippo Bonazzi (editor)

#CVE #D-BUS

Table of Contents

1) Introduction

In recent releases of the KDE desktop environment a fork of the SDDM display manager called plasma-login-manager has been integrated. As usual this led to a review in our team for the

privileged D-Bus components contained in the package. While most of the code remains the same, the new upstream added a privileged D-Bus helper called `plasmaloginauthhelper`, which suffers from defense-in-depth security issues. The full details of the issues will be discussed in the following sections.

For this review we looked into plasma-login-manager version 6.6.2.

2) Helper Overview

`plasmaloginauthhelper` makes the D-Bus interface “org.kde.kcontrol.kcmplasmalogin” accessible to all users in the system via the D-Bus system bus. It offers three actions `sync()`, `reset()` and `save()` which are all by default protected by Polkit’s `auth_admin` setting.

These methods allow to manage configuration data stored in the home directory of the `plasmalogin` service user, which has a preset of `/var/lib/plasmalogin`. The helper runs with full root privileges and interprets various client-supplied data. The `plasmalogin` home directory has the following permissions:

```
drwxr-x--- 5 plasmalogin plasmalogin 4.0K Mar 24 13:25
```

Actually this helper is also a kind of fork of a helper found in the `sddm-kcm` repository, which we covered in a previous report. It seems the codebase has not improved since then, but rather additional attack surface has been added in the meantime.

3) Security Issues

3.a) Arbitrary `chown()` via Symlink Attack in `sync()` Method

In the `sync()` method the helper service naively performs `chown()` calls on files located in the service user’s home directory (`/var/lib/plasmalogin`), allowing a `plasmalogin` to `root` exploit.

The `chown()` is performed for the paths `$PLASMALOGIN_HOME/.config`, `$PLASMALOGIN_HOME/.config/fontconfig` as well for a list of configuration files like `plasmarc` placed into `$PLASMALOGIN_HOME/.config`.

A compromised `plasmalogin` service account can place symbolic links here to direct the `chown()` to arbitrary files in the system. After the `chown()` the helper writes client-supplied content into these files, which will also end up in arbitrary files in case of a symlink attack.

This method's logic would also allow deletion of certain files like `plasmarc` in arbitrary directories, would the relevant statement in the service implementation not lack the final filename component in the path construction:

```
QFile(homeDir + QStringLiteral("/.config/")).remove();
```

Thus this removal logic doesn't work at all at the moment, since it attempts to remove the `.config` directory instead of the actual configuration files.

3.b) Arbitrary File Deletion in `reset()` Method

In the `reset()` method the paths `$PLASMALOGIN_HOME/.cache` and `$PLASMALOGIN_HOME/.config/fontconfig` are recursively deleted. For this purpose the Qt API `QDir::removeRecursively()` is used. The implementation of this function follows symbolic links even in the final path component, which means that a compromised `plasmalogin` service user can leverage this logic to achieve the deletion of arbitrary directory trees in the system.

3.c) Symlink Attack via `/var/lib/plasmalogin/wallpapers` in `save()` Method

In the `save()` method the path `/var/lib/plasmalogin/wallpapers` is created and opened by root, using a system call sequence affected by a race condition. A compromised `plasmalogin` user can replace the directory by a symbolic link in time for the service to write wallpaper files to arbitrary locations in the system, leading to local Denial-of-Service (DoS) and integrity violation.

In this spot the helper employs a low-level `openat2()` system call to avoid symbolic link resolution, but this only applies to the actual files placed within the wallpaper directory, not to the directory itself, which is naively opened before that.

3.d) Missing Integrity Check of Configuration Data in `save()` Method

In the `save()` method the contents of the file `/etc/plasmalogin.conf` can be completely controlled by the caller. Since this method is protected by `auth_admin` Polkit authentication this is basically acceptable, but there is not even an integrity or syntax check of the data, the method blindly forwards whatever the client passes to it into this file, without a maximum size limit or any sanity checks. While this is not directly a security issue it is a lack of robustness,

because the D-Bus service is responsible for maintaining a sane structure of the privileged configuration file, preventing a broken system e.g. in case of buggy clients.

3.e) Lack of File Descriptor and File Size Verification in `save()` Method

For the actual wallpaper files, file descriptor passing is employed, which is good. There is no upper limit enforced on the amount of data placed into the wallpapers directory, however, which allows to exhaust disk space in `/var/lib/plasmalogin`.

Even file descriptors passed from clients should be verified to check whether they refer to regular files and have no unexpected file flags set. This verification is missing.

4) Suggested Fixes

We suggested the following fixes to upstream:

- Foremost the helper should drop privileges to the `plasmalogin` user before performing any file system operations in `/var/lib/plasmalogin`, thereby eliminating all symlink attack surface. There still remains Denial-of-Service (DoS) attack surface if the service user places e.g. a named FIFO pipe somewhere. Avoiding this requires careful inspection of each path component by the service before opening it.
- The helper should verify the structure and size of data written to `/etc/plasmalogin.conf`.
- The helper should place a limit on the maximum amount of space which may be used for wallpapers in the `plasmalogin` user's home directory.
- The helper should verify the type and flags of file descriptors passed by the client. The descriptors should not have special file types and they should not have any unexpected flags like `O_PATH` set.

5) Severity

None of the issues in this report is exploitable in a default installation of `plasma-login-manager`. Most of the problems affect the situation when the `plasmalogin` service user is compromised and thus affect defense-in-depth.

It is conceivable, however, that some actions in the helper, like the wallpaper management, could be reduced to lesser authentication requirements like a Polkit `yes` setting for locally

logged-in users in the future, be it due to upstream changes or due to choices made by system integrators. Then further problems like disk space exhaustion by other unprivileged users could sneak in as well.

Based on the high severity of the defense-in-depth issues shown in this report, our assessment is that there is effectively no separation between `root` and the `plasmalogin` service user account.

6) Upstream Bugfix

At this time there is no bugfix available by upstream, but a security fix is planned for the next Plasma release on May 12. We have not been involved in upstream's bugfix process so far and have no knowledge about the approach that will be taken to address the issues from this report.

7) CVE Assignment

We suggested a single CVE assignment relating to the lack of privilege drop of the D-Bus service, which is the root cause of most of the issues described in this report. In coordination with upstream we assigned CVE-2026-25710 and shared it with them to track these defects.

8) Timeline

2026-03-30	We reached out to security@kde.org with a report of the problems, offering coordinated disclosure. We stated that in our eyes, due to the issues being restricted to defense-in-depth, an embargo would not be strictly necessary.
2026-03-30	Upstream provided a short reply, asking for a CVE assignment.
2026-03-31	We assigned CVE-2026-25710 and shared it with upstream.
2026-04-13	Lacking a more detailed response from upstream we asked once more whether they would like to perform coordinated disclosure and what the desired coordinated release date (CRD) would be in that case.

2026-04-13	We received a reply from upstream stating that no coordinated disclosure would be necessary and bugfixes would be published via public pull requests soon in expectation of a security release on May 12.
2026-04-13	To be sure we asked upstream once more whether they agreed to us publishing the report right away.
2026-04-20	Lacking a response and with no visible publication on upstream's end we asked once more if publication on our end would be acceptable for them.
2026-04-21	We received a response confirming that we were allowed to publish right away.

9) References

- blog post about issues in sddm-kcm6
- blog post about the introduction of file descriptor passing in kauth
- openSUSE plasma-login-manager KCM helper review bug
- plasmaloginauthhelper code
- plasma-login-manager repository