Runtime Infrastructure - Bug #12104

Patching NCURSES (6.6) Using Static Linking on Ubuntu 26.04 LTS

11/02/2026 08:33 AM - André van Hulzen

Status:	New	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Roger Borrello	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**reviewer:**	
vendor_id:	GCD	**production:**	No
case_num:		**env_name:**	
version_reported:		**topics:**	
version_resolved:			

Description

History

#1 - 11/02/2026 08:36 AM - André van Hulzen

We are planning our anual third party upgrade. We did some testing with FWD on Ubuntu 26.04. It has ncurses 6.6, but the patches we cannot be applied to ncurses 6.6. We would like patches compatible with ncurses 6.6.

#2 - 11/02/2026 09:21 AM - Greg Shah

- Assignee set to Roger Borrello

#3 - 11/02/2026 09:44 AM - André van Hulzen

Appologies. We do not need it. We are using PTHREADS these days.

#4 - 11/02/2026 09:59 AM - Roger Borrello

André van Hulzen wrote:

> Appologies. We do not need it. We are using PTHREADS these days.

So this is to be rejected, or should we patch in perpetuity? In other words, are we still supporting dynamic linking or just static linking?

#5 - 11/02/2026 10:14 AM - Greg Shah

Yes, we still support dynamic linking.

#6 - 11/02/2026 05:40 PM - Roger Borrello

André, the failure you saw comes from the v6.1 patches, which tools/scripts/setup_ncurses6x.sh *always* applied before trunk **r16699** (and which the older wiki steps use). They do not apply to 6.6. The v6.4 patches *do* apply cleanly to the 26.04 ncurses 6.6 source, and the newer script picks them automatically. If you ever need the patched build (-f), use setup_ncurses6x.sh from trunk **r16699** or later (preferably the 11678a version once merged) since it recognizes 26.04 (see #11515-44). With PTHREADS (the default, no -f) no patch is involved at all.

Greg, on dynamic linking, here is what I found (and Eugenie, I apologize if you already tried to communicate this and I missed it):

- What removes the need for auto_getch_refresh() is thread safety, not static linking. With --with-pthread, ncurses serializes wrefresh() with an internal mutex, which covers both the getch() refresh and screen output (#2660-19). A shared libncursest.so should behave the same, but only the static build was regression-tested (#2660-22, #2660-23).
- Ubuntu does not ship a thread-safe ncurses (#2660-25), so a shared thread-safe setup means building and installing libncursest.so.6 and libtinfot.so.6 ourselves. They have their own sonames, so they can sit beside the system library (e.g. in /opt/ncurses/lib) without replacing it.
- The dynamic path in the makefile has no protection today. Without NCURSES_FWD_STATIC it links the stock -lncurses, which is not thread-safe. Since **r15728**, auto_getch_refresh(FALSE) in terminal_linux.c is only compiled in for the static non-pthreads build (NCURSES_FWD_USE_PATCH), so it is skipped even when the system library is patched. That is the combination that produced screen corruption in #2660-22.

Options (not mutually exclusive):

1. Support shared thread-safe: a shared-library option in setup_ncurses6x.sh, a makefile branch linking -lncursest -ltinfot -lpthread, and a ChUI regression run against it.
2. Restore the patched dynamic path: define NCURSES_FWD_USE_PATCH for the -lncurses branch again, as before r15728.
3. Drop dynamic linking and support static only.

Greg, Eugenie: which way do you want to go?

#7 - 11/02/2026 06:06 PM - Eugenie Lyzenko

Roger Borrello wrote:

> André, the failure you saw comes from the v6.1 patches, which tools/scripts/setup_ncurses6x.sh *always* applied before trunk **r16699** (and which the older wiki steps use). They do not apply to 6.6. The v6.4 patches *do* apply cleanly to the 26.04 ncurses 6.6 source, and the newer script picks them automatically. If you ever need the patched build (-f), use setup_ncurses6x.sh from trunk **r16699** or later (preferably the 11678a version once merged) since it recognizes 26.04 (see #11515-44). With PTHREADS (the default, no -f) no patch is involved at all.
>
> Greg, on dynamic linking, here is what I found (and Eugenie, I apologize if you already tried to communicate this and I missed it):
>
> - What removes the need for auto_getch_refresh() is thread safety, not static linking. With --with-pthread, ncurses serializes wrefresh() with an internal mutex, which covers both the getch() refresh and screen output (#2660-19). A shared libncursest.so should behave the same, but only the static build was regression-tested (#2660-22, #2660-23).
> - Ubuntu does not ship a thread-safe ncurses (#2660-25), so a shared thread-safe setup means building and installing libncursest.so.6 and libtinfot.so.6 ourselves. They have their own sonames, so they can sit beside the system library (e.g. in /opt/ncurses/lib) without replacing it.
> - The dynamic path in the makefile has no protection today. Without NCURSES_FWD_STATIC it links the stock -lncurses, which is not thread-safe. Since **r15728**, auto_getch_refresh(FALSE) in terminal_linux.c is only compiled in for the static non-pthreads build (NCURSES_FWD_USE_PATCH), so it is skipped even when the system library is patched. That is the combination that produced screen corruption in #2660-22.
>
> Options (not mutually exclusive):
>
> 1. Support shared thread-safe: a shared-library option in setup_ncurses6x.sh, a makefile branch linking -lncursest -ltinfot -lpthread, and a ChUI regression run against it.
> 2. Restore the patched dynamic path: define NCURSES_FWD_USE_PATCH for the -lncurses branch again, as before r15728.
> 3. Drop dynamic linking and support static only.
>
> Greg, Eugenie: which way do you want to go?

I would prefer to use only static library which we have full control for building. This gives som independency on the system wide unpredictable changes/tunes/improvements. As far as I know since some revision the NCURSES will be deployed with Wide char capable version (w suffix for .so

name). And we can not predict what will be the next change tomorrow. So I choose stability in this area.