Debian Official Debug Kernel for All?

Yunseong Kim <ysk@kzalloc.com>
www.kzalloc.com

東京 2025
LINUX
PLUMBERS CONFERENCE
TOKYO, JAPAN / DEC. 11-13, 2025

Community based Enterprise Distributions

Community based Enterprise distributions have recognized debug kernels as essential infrastructure.

[Fedora & RHEL]

Provides official kernel-debug packages with comprehensive sanitizer support: ARM64, X86_64, RISCV64, and S390

KASAN, UBSAN, concurrency, and memory debugging enabled by default.

[Open SUSE & Enterprise]

Maintains dedicated debug configurations in their kernel-source repository.

ARM64 and x86_64 debug variants available with extensive memory checking and lock validation features enabled.

debian

Currently provides debug symbols (dbgsym)

No official sanitizer enabled kernel images.

Developers must build custom kernels.

My patch work for Debian Debug Kernel

Debug Configurations Across Distributions?

Performance

Enabling certain Sanitizers introduces significant overhead,
rendering the system unsuitable for real-world workloads.

Reproducibility

We failed to reproduce hardware-specific bugs observed in production.
Additionally, generic debug configurations can sometimes yield false positives.

Proposed "Common Ground" (Lightweight Reference)

Inspired by industry standards (e.g., Qualcomm Debug Kernel Configuration),
I propose a minimum viable debug set to align with other distros.

Debug Configurations Across Distributions?

The "Lightweight" Checklist

Kernel debug configuration option	Description
CONFIG_DEBUG_LIST	Turns on checks for performing standard linked list manipulations with the *list.h* header file. If the pointers don't match, the system prints a warning, followed by a BUG_ON crash.
CONFIG_PAGE_POISONING	Fills the pages with the poison pattern (PAGE_POISON 0xaa), after calling free_pages().
CONFIG_DEBUG_PAGEALLOC	Verifies the patterns before calling alloc_pages().
CONFIG_DEBUG_USER	Prints a message when the system kills a user-space process due to a segmentation fault (segfault) or an invalid instruction, such as user_debug=31 in the *arch/arm/Kconfig.debug* file. Add the Kernel boot parameter to the *BoardConfig.mk* file.
CONFIG_DEBUG_SPINLOCK	Identifies missing spinlock initialization and common spinlock errors, such as: • Waiting for more than one second on a spinlock • Freeing an already freed lock • Re-initializing a lock that was already used
CONFIG_DEBUG_MUTEXES	Detects Mutex semantic violations.
DEBUG_LOCK_ALLOC	Detects wrong freeing of live locks.
CONFIG_SLUB_DEBUG	Performs extra checks to detect the corruption of internal kernel memory allocation structures by adding poison for use-after-free (0x6b) and buffer-overflow-padding (0xbb).

https://docs.qualcomm.com/doc/80-70018-12/topic/debugging_linux_kernel.html#debugging-linux-kernel

Debug Configurations Across Distributions?

The "Middleweight" Checklist

Kernel debug configuration for extra debugging can be verbose and can make the system slow.

Kernel debug configuration option	Description
CONFIG_DEBUG_ATOMIC_SLEEP	Causes routines that may sleep to become noisy when they're called within the atomic sections.
DEBUG_SPINLOCK_SLEEP	Causes routines that may sleep to become noisy when they're called with a spinlock held.
CONFIG_DEBUG_VM, CONFIG_DEBUG_HIGHMEM	Provides an extra debugging support for virtual memory management corruption issues.
CONFIG_DEBUG_OBJECTS	Tracks the lifetime of various objects and validates the operations on those objects.

https://docs.qualcomm.com/doc/80-70018-12/topic/debugging_linux_kernel.html#debugging-linux-kernel

The Pain Point

The "More Info Needed" Loop

User Report: "System freezes randomly" or "Reboots unexpectedly."
Team Request: "Can't reproduce. Please send another information."

The Bottleneck:
Building a kernel is a high barrier for most users.
https://bugs.debian.org/cgi-bin/pkgreport.cgi?src=linux
The user gives up. The ticket is marked more info and becomes stale.

Critical bugs remain hidden and unresolved.

The Debian Delta

Debian-Specific Patches

Upstream CI covers the mainline kernel thoroughly.
But how much visibility does debian/patches/*?

Debian carries numerous custom patches
(e.g., Secure Boot integration, VFS changes, legacy hardware support).

Such changes carry a risk of unintended consequences, such as race conditions or memory corruption, which fall outside the scope of upstream CI.

Manual verification has inherent limitations.

The Solution: Official Debug Flavors

We can make Debian Kernel Package

Introduce official debug-amd64 and debug-arm64 flavours in experimental (and later sid).

Define new flavors in **defines.toml** & Adding **config.debug**:

Memory Safety: Detect out-of-bounds, use-after-free
CONFIG_KASAN=y, CONFIG_UBSAN=y

Concurrency: Detect deadlocks with Lockdep
CONFIG_PROVE_LOCKING=y

Coverage: Enable coverage-guided kernel execution paths triggered by user applications.
CONFIG_KCOV=y

What application do on debug kernel

KCOV and debug kernels are only for fuzzers like syzbot.

vock (Visual Kernel Coverage)

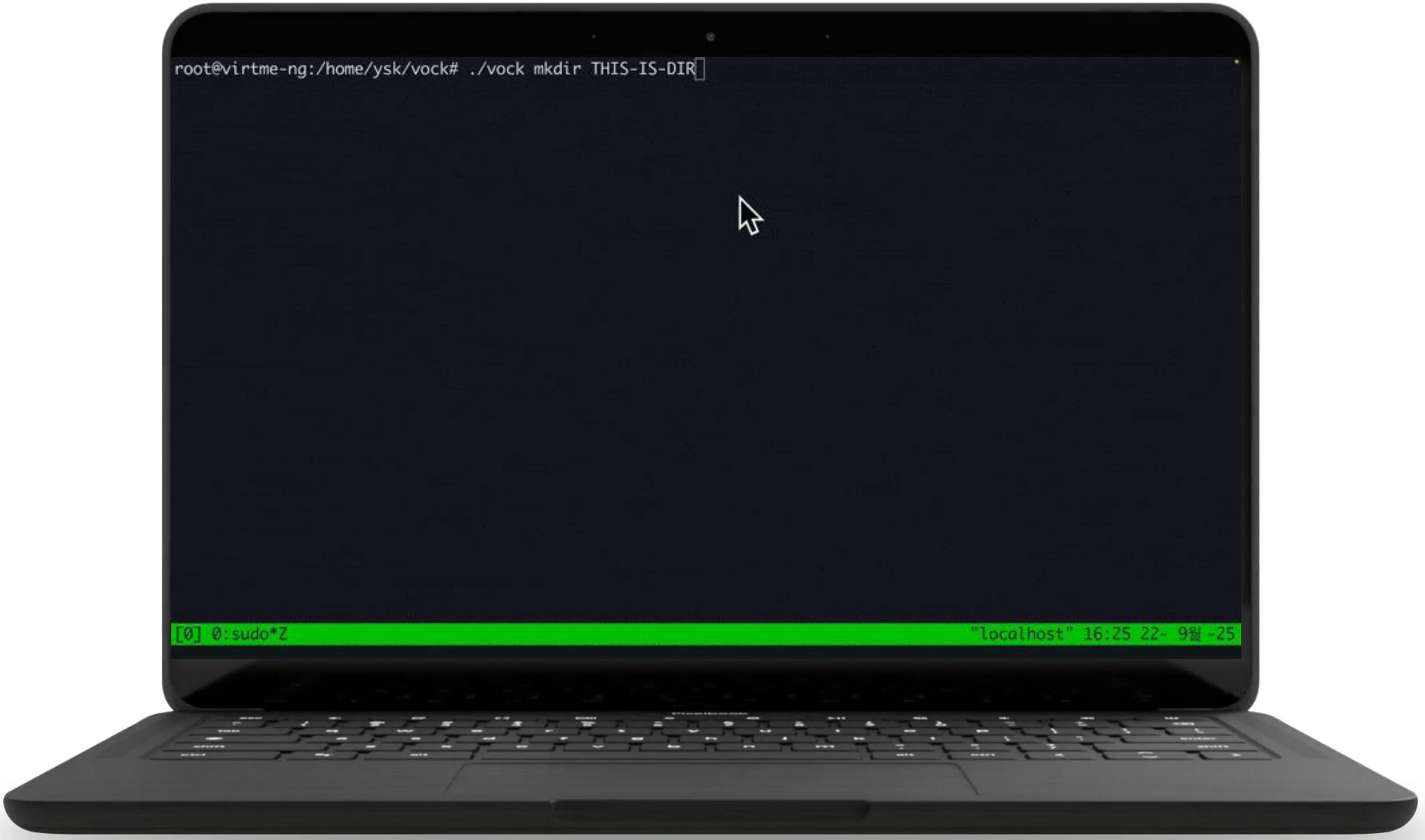
Problem: Text logs (dmesg) are often insufficient to understand complex application behavior.

Solution: Use KCOV to record the kernel execution path triggered by a specific application.

Result: Visualize the execution flow as a heatmap on the source code.

"Stop guessing, start seeing."

With official debug kernels, any Debian user can apt install and use tools like vock immediately.



https://github.com/kzall0c/vock

Security Ecosystem: syzbot the Kernel Bug Hunter

Google's syzbot team has expressed support for Debian.

We can provide the debug kernel images/headers.

Proactive QA: Syzbot validates Debian-specific patches against corner cases we can't manually test.

High-Quality Reports: Actionable reproducers (C programs) + Root Cause Analysis.

Join the ecosystem alongside Upstream, Android.

Debian kernel testing on syzbot

조회수 27회

Dmitry Vyukov

받는사람

Hello,

Our team works on syzkaller/syzbot kernel fuzzer:
https://github.com/google/syzkaller
https://github.com/google/syzkaller/blob/master/docs/syzbot.md

Currently we test the upstream kernel and recent LTS releases:
https://syzkaller.appspot.com/upstream
https://syzkaller.appspot.com/linux-6.1
https://syzkaller.appspot.com/linux-5.15
and report bugs to upstream developers:
https://groups.google.com/g/syzkaller-bugs

Due to Debian's relevance as one of the most widely used Linux distributions, we plan to test the Debian kernel as well.

We were thinking about testing the "testing" release only initially.
Or do you have other suggestions here?

Do you want bugs to be reported privately first (to some closed mailing list) with some embargo? Or do we make them public (visible on syzbot dashboard) right away as we do for upstream/LTS?

Ben Hutchings

받는사람

On Wed, 2023-06-28 at 10:26 +0200, Dmitry Vyukov wrote:
> On Thu, 22 Jun 2023 at 16:46, Dmitry Vyukov <dvy...@google.com> wrote:
> >
> > Hello,
> >
> > Our team works on syzkaller/syzbot kernel fuzzer:
> > https://github.com/google/syzkaller
> > https://github.com/google/syzkaller/blob/master/docs/syzbot.md
> >
> > Currently we test the upstream kernel and recent LTS releases:
> > https://syzkaller.appspot.com/upstream
> > https://syzkaller.appspot.com/linux-6.1
> > https://syzkaller.appspot.com/linux-5.15
> > and report bugs to upstream developers:
> > https://groups.google.com/g/syzkaller-bugs
> >
> > Due to Debian's relevance as one of the most widely used Linux
> > distributions, we plan to test the Debian kernel as well.
> >
> > We were thinking about testing the "testing" release only initially.
> > Or do you have other suggestions here?

If you want to find issues affecting the next release, then that's the right choice. But if you want to find issues that still need fixes uploaded, then "unstable" is the right choice. Any fixes in testing need to go via unstable.

> > Do you want bugs to be reported privately first (to some closed
> > mailing list) with some embargo? Or do we make them public (visible on
> > syzbot dashboard) right away as we do for upstream/LTS?
>
> +Ben, you were pointed out as the person to provide "the official" response :)

I'm just one person on the kernel team, and not the most active at the moment. Salvatore Bonaccorso is doing most of the security updates.

> To clarify: we are not asking nor imply that anybody will actually act
> in any way on the reported bugs. I mean anybody is welcome to, but
> don't have to.
> We can also just create a public web dashboard (+new opt-in mailing
> list), if that's what we agree on here.
>
> And if there is an active interest in acting on the reports, we can
> also test the unstable release (that's the better place to fix,
> right).

If syzbot is able to distinguish bugs that are reproducible on Debian patched kernels but not in the corresponding stable releases, I think that would be very useful to us. My guess is that this would be a manageable rate of bugs and we could receive those privately. What do you think, Salvatore?

If this isn't possible, then it's unlikely we will have the time to look at the issues. You can create a public web dashboard but I don't know if that's going to help anyone.

Ben.

https://groups.google.com/g/linux.debian.kernel/c/9MKP0aljQrE

Salvatore Bonaccorso

받는사람

Hi,

•••

Yes, I do agree with the above. If syzbot can do that separation then it would be useful, and think we can agree to get those reports privately to either a deidcated set up distribution list or directly to the Debian maintainers. In first stance I guess that would be Ben and me, and possibly the others listed as Uploaders for the src:linux package.

OTOH, I fully agree, if syzbot canot distinguish issues reproducible only with Debian patches applied I think we won't really have the free time to investigate those reports. As we are doing this in our free time.

Thank you for approaching us on this!

Regards,
Salvatore

Ben Hutchings

On Wed, 2023-06-28 at 10:26 +0200, Dmitry Vyukov wrote: > On Thu, 22 Jun 2023 at 16:46, Dmitry

Salvatore Bonaccorso

Hi, On Sun, Jul 16, 2023 at 07:47:11PM +0200, Ben Hutchings wrote: > On Wed, 2023-06-28 at 10:26 +

Dmitry Vyukov

받는사람

•••

Hi Ben, Salvatore,

Yes, syzbot can detect "downstream" bugs with reasonable precision,
e.g. for Android 5.15 tree:
https://syzkaller.appspot.com/android-5-15?label=origin%3Adownstream
and here you can see "Bug presence" analysis by testing on
corresponding LTS and upstream trees:
https://syzkaller.appspot.com/bug?extid=ea487d1ec1a25689e4d2

However, I am not sure if we can readily set up reporting of only such bugs privately.

https://groups.google.com/g/linux.debian.kernel/c/9MKP0aljQrE

Proactive Security

From Reactive to Proactive

Wait for CVE -> Backport fix -> Release.

Proactive Goal:
Detect bugs via Fuzzing -> Fix silently -> Release secure kernel.

Memory safety issues account for > 70% of vulnerabilities.

Catching them at the "Bug" stage prevents them from becoming "Exploits."

Roadmap & Challenges

with Proposed Roadmap

Challenges: Many architecture x Various page sizes supported by ARM64 x K*SAN

Build Time: ~3x increase due to sanitizer instrumentation.

Storage: Larger package size (~2.5GB per flavor).

Phase 1: Pilot & Assessment - Release debug flavors in the experimental repo to assess infrastructure impact.

Phase 2: CI Automation - Incorporate boot tests with sanitizers into Salsa CI pipeline.

Phase 3: Ecosystem Expansion - Extend coverage by integrating with syzbot & KernelCI.

Conclusion & Call to Action

Let's discuss flavour.debug official in Debian

Reduce Issue Cost: Turn "unreproducible" bugs into actionable reports.

Improve Quality: Verify Debian-specific patches automatically.

Useful Tools for debug kernel

Debug kernels are not a waste of resources;
They are an investment in maintainability and security.

Q&A

東京 2025
LINUX
PLUMBERS CONFERENCE
TOKYO, JAPAN / DEC. 11-13, 2025

References

Debian Kernel Handbook

https://kernel-team.pages.debian.net/kernel-handbook/

Fedora & RHEL Debug Kernel Config

https://src.fedoraproject.org/rpms/kernel/tree/rawhide

Open SUSE Debug Kernel Config

https://github.com/openSUSE/kernel-source/blob/SUSE-2024/config/arm64/debug

Qualcomm Debug Kernel Config

https://docs.qualcomm.com/doc/80-70018-12/topic/debugging_linux_kernel.html

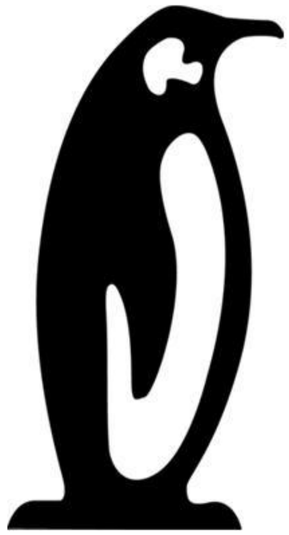
syzkaller debug config for the android

https://github.com/google/syzkaller/blob/master/dashboard/config/linux/android-6.12.config

Kernel CVEs are Alive, but Do Not Panic!
Greg Kroah-Hartman, Kernel Maintainer

https://youtu.be/dhu8HSOzxd8

東京 2025
LINUX
PLUMBERS CONFERENCE
TOKYO, JAPAN / DEC. 11-13, 2025



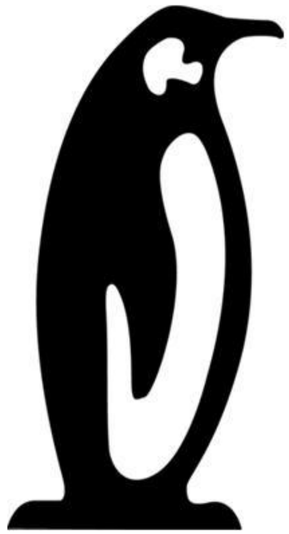
東京 2025
LINUX PLUMBERS CONFERENCE

TOKYO, JAPAN / DECEMBER 11-13, 2025

Kernel debug configuration option	Description
CONFIG_DEBUG_LIST	This option turns on checks for performing standard linked list manipulations with the *list.h* header file. If the pointers don't match, the system prints a warning, followed by a BUG_ON crash.
CONFIG_PAGE_POISONING	This option fills the pages with the poison pattern (PAGE_POISON 0xaa), after calling free_pages().
CONFIG_DEBUG_PAGEALLOC	This option verifies the patterns before calling alloc_pages().
CONFIG_DEBUG_USER	This option prints a message when the system kills a user-space process due to a segmentation fault (segfault) or an invalid instruction, such as user_debug=31 in the *arch/arm/Kconfig.debug* file. Add the Kernel boot parameter to the *BoardConfig.mk* file.
CONFIG_DEBUG_SPINLOCK	This option identifies missing spinlock initialization and common spinlock errors, such as: • Waiting for more than one second on a spinlock • Freeing an already freed lock • Reinitializing a lock that was already used
CONFIG_DEBUG_MUTEXES	This option detects Mutex semantic violations.
DEBUG_LOCK_ALLOC	This option detects wrong freeing of live locks.
CONFIG_SLUB CONFIG_SLUB_DEBUG	This option performs extra checks to detect the corruption of internal kernel memory allocation structures by adding poison for use-after-free (0x6b) and buffer-overflow-padding (0xbb).
Kernel debug configuration options for extra debugging that can be verbose and can make the system slow.	
CONFIG_DEBUG_ATOMIC_SLEEP	This option causes routines that may sleep to become noisy when they're called within the atomic sections.
DEBUG_SPINLOCK_SLEEP	This option causes routines that may sleep to become noisy when they're called with a spinlock held.
CONFIG_DEBUG_VM, CONFIG_DEBUG_HIGHMEM	This option provides an extra debugging support for virtual memory management corruption issues.
CONFIG_DEBUG_OBJECTS	This option tracks the lifetime of various objects and validates the operations on those objects.

https://docs.qualcomm.com/doc/80-70018-12/topic/debugging_linux_kernel.html#debugging-linux-kernel


東京 2025
LINUX
PLUMBERS CONFERENCE
TOKYO, JAPAN / DEC. 11-13, 2025



東京 2025
LINUX PLUMBERS CONFERENCE

TOKYO, JAPAN / DECEMBER 11-13, 2025

LPC 2023 - Overview

Conference Details

The Linux Plumbers Conference is the premier event for developers working at all levels of the plumbing layer and beyond.

Taking place on Thursday 11th, Friday 12th and Saturday 13th of December, this year we will be both in person and remote (hybrid). However to minimize technical issues, we'd appreciate most of the content presenters being in-person.

The in-person venue is the Toranomon Hills Forum, Tokyo, Japan

Toranomon Hills Mori Tower 5th Floor, 1-23-3 Toranomon, Minato-ku, Tokyo, 105-6305, Japan

Unless specified otherwise, the conference information will be shared in Japan (JST timezone).

Sponsorship opportunities

Linux Plumbers Conference would not be possible without our sponsors. Many thanks to all the great organizations that have supported Linux Plumbers Conference over the years.

New sponsorship opportunities are available for 2025! We hope that your organization will consider joining our growing list of amazing sponsors this year. Find out more here.

東京 2025
LINUX
PLUMBERS CONFERENCE
TOKYO, JAPAN / DEC. 11-13, 2025

LPC 2023 - Overview

Conference Details

The Linux Plumbers Conference is the premier event for developers working at all levels of the plumbing layer and beyond.

Taking place on Thursday 11th, Friday 12th and Saturday 13th of December, this year we will be both in person and remote (hybrid). However to minimize technical issues, we'd appreciate most of the content presenters being in-person. Taking place on Thursday 11th, Friday 12th and Saturday 13th of December, this year we will be both in person and remote (hybrid). However to minimize technical issues, we'd appreciate most of the content presenters being in-person.

The in-person venue is the Toranomon Hills Forum, Tokyo, Japan

Toranomon Hills Mori Tower 5th Floor, 1-23-3 Toranomon, Minato-ku, Tokyo, 105-6305, Japan

Unless specified otherwise, the conference information will be shared in Japan (JST timezone).

Conference Details

The Linux Plumbers Conference is the premier event for developers working at all levels of the plumbing layer and beyond.

東京 2025
LINUX
PLUMBERS CONFERENCE
TOKYO, JAPAN / DEC. 11-13, 2025