Contribution ID: **119** Type: **not specified**

Debian Official Debug Kernel for All?

Thursday 11 December 2025 10:30 (30 minutes)

Problem Statement

The security and stability of the Linux kernel are paramount to the entire open-source ecosystem. A critical component in achieving this is the availability of debug kernels—builds specifically enabled with intensive debugging features like KASAN, UBSAN, and other sanitizers. While enterprise distributions such as RHEL, Fedora, and SUSE rely heavily on official kernel-debug packages as a cornerstone of their security strategies, Debian—one of the most influential community distributions—does not.

As of 2025, Debian provides valuable debug symbols (dbgsym). However, an officially supported, sanitizer-enabled kernel image is not yet available. This presents an opportunity to further enhance proactive hardening of the Debian kernel and improve the detection of complex vulnerabilities. While the current practice of developers building their own debug kernels is a functional workaround, it introduces hurdles for consistency and collaborative debugging at the scale of the Debian project:

1. **Inconsistent Security Research and Bug Reproduction:** Effective security research and collaborative debugging depend entirely on standardized, reproducible binaries. Manual builds lack consistency; variations in build environments make it difficult to reliably reproduce and verify security vulnerabilities or complex bugs, slowing down the patching process.
2. **Compromised Chain of Trust and Incident Response:** In security-critical environments and Secure Boot setups, deploying unsigned, manually built kernels is often prohibited. This leaves administrators unable to utilize advanced debugging tools when they are needed most, such as during incident response. Official, signed debug kernels ensure the integrity of the Debian chain of trust.
3. **Inefficient Resource Allocation:** The significant time spent compiling custom kernels detracts from the actual work of analyzing crashes, debugging, and patching vulnerabilities.
4. **A Barrier to Automated Security Testing (Syzbot):** A significant consequence is the inability to integrate Debian kernels into Syzbot, the automated kernel fuzzing system responsible for finding thousands of vulnerabilities upstream. Syzbot relies on sanitizer-enabled kernels to detect memory corruptions, race conditions, and other critical flaws. By not providing a compatible debug kernel, Debian misses out on continuous, automated security auditing. This omission delays fixes for Debian users and hinders Debian's ability to contribute unique bug findings (especially across its diverse architectures) back to the upstream kernel.

We assert that providing an official debug kernel is no longer optional but a necessity for Debian to actively contribute to the Linux kernel's security and stability, moving beyond reactive fixes to proactive detection.

The Core Challenge: Infrastructure Constraints

The Debian Kernel Team's cautious stance is understandable. Debian operates as a non-profit, community-driven project, supporting a vast array of architectures (e.g., amd64, arm64, mips, riscv) across multiple release channels (stable, testing, unstable).

Adding a debug kernel flavor for every combination (Architecture × Kernel Version) means an exponential increase in the number of packages, imposing severe strain on the infrastructure:

- **Build Time:** The capacity of the automated build infrastructure (buildd network) is finite, and enabling sanitizers significantly increases compilation time.
- **Storage Requirements:** The archive size would grow substantially, increasing storage costs.
- **Mirror Network Load:** Distributing these large packages across the global mirror network generates significant traffic, a burden that Debian, unlike corporate-backed distributions, finds difficult to bear.

Proposed Discussion Topics

The goal of this session is to explore **sustainable implementation strategies** that enable proactive security testing (like Syzbot integration) while respecting Debian's infrastructure constraints. We invite build system experts and maintainers from other distributions to discuss the following topics:

1. **Mitigation Strategies for Infrastructure Load:**
   - Could a phased rollout be a realistic compromise? For example, prioritizing only amd64/arm64 and the unstable channel initially?
   - How effective would it be to distribute debug kernel packages through a separate repository (e.g., debug-archive) to reduce the burden on global mirrors?
2. **Build System Optimization and Integration:**
   - How can we efficiently integrate a new debug flavor into the existing Debian kernel build pipelines?
   - What optimization strategies (e.g., caching strategies, artifact reuse) can be employed to reduce the build time of sanitizer-enabled kernels?
3. **Lessons Learned from Other Distributions:**
   - How do RHEL, Fedora, and SUSE manage the build and storage overhead associated with maintaining kernel-debug packages?
   - What build infrastructure strategies from their experience can Debian adopt?
4. **Defining the Minimum Viable Product (MVP) for Security:**
   - Aiming specifically for Syzbot integration and security auditing, what is the minimal debug kernel configuration required to maximize the security impact relative to the infrastructure investment?

Desired Outcomes

Through this discussion, we aim to develop a concrete, phased proposal for introducing an official debug kernel in Debian, prioritizing security benefits. Specifically, we aim to identify technical strategies from a build system perspective to overcome infrastructure constraints and define the next steps for a Proof of Concept (PoC).

Preparation Materials

Attendees are encouraged to review the following materials to understand the background and participate productively in the discussion:

- Debian Kernel Team meeting minutes regarding debug kernel (2025-08-13): https://salsa.debian.org/kernel-team/meetings/-/wikis/20250813#note_639360
- Mailing list discussion on Debian debug kernel support and syzbot: https://groups.google.com/g/linux.debian.kernel/c/9MKPOaljQ

Primary author: KIM, Yunseong (Ericsson)

Co-author: JIN, seonghee (Georgia Institute of Technology)

Presenters: KIM, Yunseong (Ericsson); JIN, seonghee (Georgia Institute of Technology)

Session Classification: Build Systems MC

Track Classification: Build Systems MC