A Thesis for the Degree of Ph.D. in Engineering

Improving Linux Kernel Reliability through Targeted Static Analysis

September 2025

Graduate School of Science and Technology
Keio University

Keita Suzuki

別表５
(3)

主　論　文　要　旨

No.1

報告番号	甲 乙 第　　号	氏　名	鈴木 慶汰

主 論 文 題 名 ：
Improving Linux Kernel Reliability through Targeted Static Analysis
(ターゲット特化型静的解析による Linux カーネルの信頼性向上)

（内容の要旨）

現代のコンピューティングシステムの基盤である Linux カーネルは，組み込みシステムや個人用デバイスから，証券取引所や航空管制システムのような重要なインフラに至るまで，多様な領域で広く利用されている．しかし，その広大なサイズ（3,000 万行以上のコード）と複雑さゆえに，システムの信頼性やセキュリティを損なう可能性のある多くのバグに対して脆弱である．これらのバグを検出すること，特にハードウェアへの依存を避ける静的解析による検出は極めて重要だが，静的解析の実用化には解析のスケーラビリティと精度のトレードオフという課題に直面している．

本論文は、Linux カーネルの開発ライフサイクルにおける 2 つの主要なフェーズで静的解析の要件が大きく異なることを認識し，この課題に取り組む．本研究では，速度重視の日々の開発者ワークフローのための実用的なフレームワークと，包括的な統合フェーズでしばしば見逃される困難なバグパターンを対象としたスケーラブルな手法という、2 つの新規かつ的を絞った静的解析アプローチを提案する．

最初の貢献は，日々の開発者ワークフローにおけるバグ検出ツールの活用不足に対処する．高機能ツールの長い解析時間がその原因であるため，本研究では高速かつワークフローに統合されたバグチェッカーを生成するフレームワーク FiTx を導入する．FiT バグとは，計算コストの低い単一コンパイル単位の解析で検出可能なバグである。Linux v5.15 において，FiTx はソースファイルの 90 %に対してわずか 0.99 秒で解析を完了し，47 件の新規バグを発見した．一般的なツールと比較して生成される警告が大幅に少なく，報告されたバグのうち 13 件は開発者によって確認された．

2 番目の貢献は，統合フェーズで見逃されがちな構造体メンバー関連のメモリリークを対象とする．このリークタイプは最近の修正パッチの 54.6 %を占め，その 77.6 %が手動で発見されており，既存ツールの見落としを示唆している．本研究では，高コストなグローバルポインティング解析を避けるため，制御フローグラフ（CFG）上で直接フィールドベースの状態追跡を用い，エラーコード解析で強化されたスケーラブルな静的解析を提案する．このツールは Linux v5.3-rc4 において，12 件の新規かつ長期にわたるバグ（真陽性警告 13 件に相当）を 59.1 %の精度で発見した．提出された全てのパッチは開発者によって確認・受理された．

総じて，本論文は静的解析の分野，および大規模なオペレーティングシステムカーネルへのその応用に対して，重要な貢献を提供する．ワークフローに統合されたバグ検出のための実用的なフレームワークと，困難なメモリリークパターンに対するスケーラブルな手法の両方を開発することにより，本研究は，対象ソフトウェアの特定の特性と開発者の運用コンテキストの両方に合わせて解析戦略を調整することの価値を実証する．

別表６
（３）

Keio University

Thesis Abstract

No.

Registration Number	□ “KOU” □ “OTSU” No. *Office use only	Name	Keita Suzuki

Thesis Title

Improving Linux Kernel Reliability through Targeted Static Analysis

Thesis Summary

The Linux kernel, a cornerstone of modern computing systems, is widely utilized across diverse domains, from embedded systems and personal devices to critical infrastructure like stock exchanges and air traffic control systems. However, its vast size (over 30 million lines of code) and complexity make it susceptible to numerous bugs, which can compromise system reliability and security. Detecting these bugs, particularly through static analysis, which avoids hardware dependencies, is crucial but faces significant challenges, notably the trade-off between analysis scalability and precision.

This dissertation addresses this challenge by recognizing that the requirements for static analysis differ significantly between the two primary phases of the Linux development lifecycle. This work proposes two novel, targeted static analysis approaches: a practical framework for the speed-focused daily developer workflow, and a scalable technique for a challenging bug pattern often missed during the comprehensive integration phase.

The first contribution addresses the underutilization of bug detection tools in the daily developer workflow, which is often caused by the long analysis times of capable tools. To solve this, we introduce FiTx, a framework that generates fast, workflow-integrated bug checkers for “Finger Traceable” (FiT) bugs. These are bugs detectable via computationally low-cost, single-compilation-unit analyses. On Linux v5.15, FiTx required only 0.99s for 90% of the source files, found 47 new bugs, and generated substantially fewer warnings than common tools, with 13 of the reported bugs confirmed by developers.

The second contribution targets a prevalent bug pattern missed during the integration phase: struct member-related memory leaks. Our study found that this leak type constitutes 54.6% of recent leak patches, with 77.6% likely being found manually, indicating they are often missed by existing scalable tools that compromise on precision. We propose a scalable static analysis that uses field-based state tracking directly on the Control Flow Graph (CFG), enhanced with error code analysis, to avoid expensive global points-to analysis. This tool found 12 new, long-lived bugs (corresponding to 13 true-positive warnings) in Linux v5.3-rc4 with 59.1% precision, and all submitted patches were confirmed and accepted by developers.

Collectively, this dissertation provides significant contributions to the field of static analysis and its application to large-scale operating system kernels. By developing both a practical framework for workflow-integrated bug detection and a scalable technique for a challenging memory leak pattern, this work demonstrates the value of tailoring analysis strategies to both the specific characteristics of the target software and the operational context of its developers.