Static Race Detection in OS Kernels by Mining Locking Rules

Abstract

To take advantage of multiple-core architecture of modern CPUs, an operating system (OS) is often designed to be highly concurrent, and processes shared data structure in parallel. To ensure safe access to shared data structures, fine-grained locking mechanisms are used to avoid data races among different threads. However, it is hard to determine whether a lock should be used for a specific data structure field (locking rules) even for an expert developer, due to poor documentation and complicated logic of OS code. As a result, OS kernel is prone to data races, because necessary locks may be missed by mistake. Static analysis is a common technique to help improve code quality, but it is quite challenging to detect data races in OS kernels automatically, because of lack of knowledge of locking rules and high complexity of concurrent execution.

In this paper, we design a practical static analysis approach named DRACE, to effectively detect harmful races that can trigger memory or logical bugs in OS kernels by mining locking rules. DRACE first employs an alias-aware rule mining method to automatically deduce locking rules, and detects data races caused by violation of these rules. And then performs a lock-usage analysis to filter out false positives caused by concurrency. At last, xxxxx extracts harmful data races from all detected data races through pattern-based estimation. We have evaluated DRACE on Linux 6.2, and find xxx data races, with a false positive rate of xxx. Among these data races, xxx are estimated to be harmful. We have reported these harmful bugs to Linux kernel developers, and xxx of them have been confirmed.

1. Introduction

References