CERIAS

The Center for Education and Research in Information Assurance and Security

Double Trouble: Fuzzing Kernel Modules

Derrick McKee, Prashast Srivastava, Hui Peng, Mathias Payer

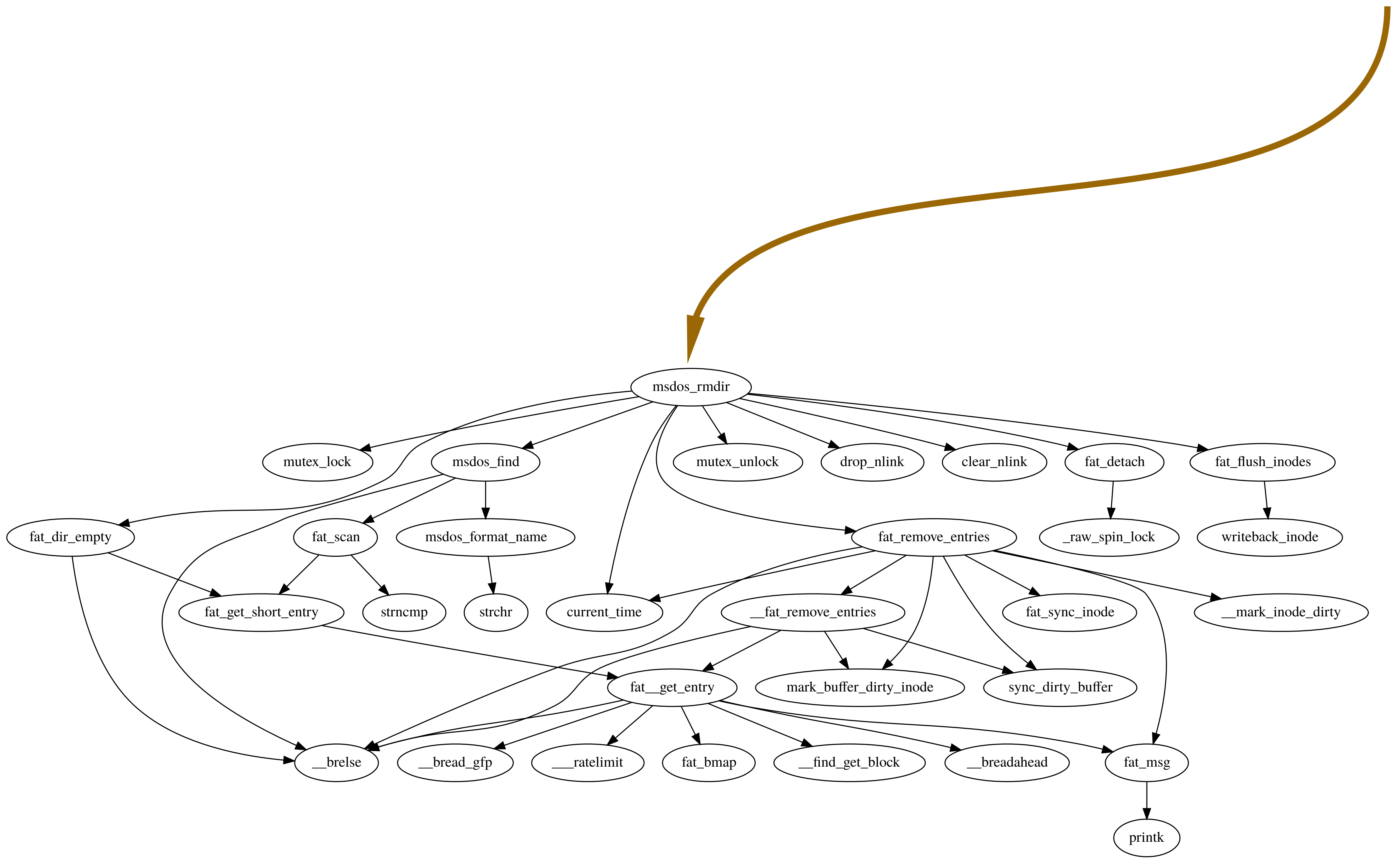
Abstract

Despite its importance, the Linux kernel remains a difficult target for fuzzers. While prior work, such as syzkaller, allows for fuzzing the kernel, it is still difficult to fuzz arbitrary loadable kernel modules (LKMs). These LKMs represent the majority of source code in the Linux kernel, as well as a major source of CVEs.

We present Double Trouble, a general framework for fuzzing LKMs. Double Trouble requires a one-time minimal programming effort to support fuzzing any subsystem. Once that effort is complete, any LKM within the subsystem can be fuzzed from the two possible sources of inputs: an upper half direction representing a system call execution, and a lower half direction representing either hardware interrupt or exposed kernel function. As a case study, we are adding support for fuzzing filesystems. We use the state-of-the-art generational fuzzer, `afl-fuzz`, for the upper half fuzzing, and a deterministic fuzzer for the lower half.

Problems

- 568 critical vulnerabilities found in the Linux kernel last year.
- Emulating a whole system drastically limits fuzzing throughput.
- The complexity of the kernel also greatly hinders the fuzzer code coverage.
- Often difficult to fuzz specific modules.



Our Solution

- Ignore kernel features that are orthogonal to correct module functionality.
- Implement “kernel gateway functions” that provide simple services, such as memory allocation.
- Provide a generic, simple API that developers can use to exercise their module functionality.
- Generate a user space application that is as fast and scalable as the underlying fuzzer.

PURDUE UNIVERSITY Discovery Park