Weenix Operating System

Adam Mroueh

May 2024

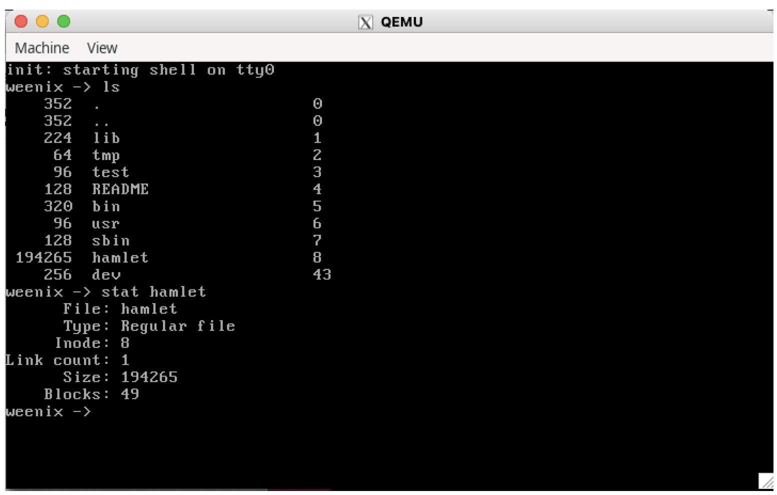
1 Overview

As part of my capstone requirement for a Bachelors of Science in Computer Science-Economics I completed the Operating Systems course (CS1670) with optional half-lab component (CS1690). Throughout the duration of this course I developed the majority of the Weenix operating system kernel through a series of five progressive assignments.

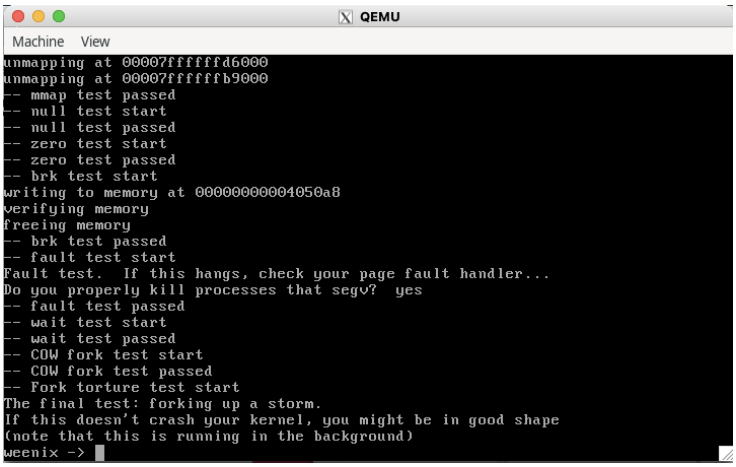
- **Processes and Threads:** I began by creating fundamental kernel structures such as threads, processes, and synchronization primitives (mutexes), enabling concurrent execution within the kernel.
- **Drivers:** I implemented device drivers for terminals, disks, and memory devices (/dev/zero and /dev/null). This included handling both block and character devices.
- **Virtual File System (VFS):** Developed a versatile interface between the Operating System and kernel and various file systems. The polymorphic design of this component emulated a Unix-style interface and allowed the kernel to handle file operations seamlessly across different file systems (ramfs, S5FS).
- **System V File System (S5FS):** I implemented the System V File System (S5FS), which is based on the original Unix file system. In this file system, each file or directory is represented by an inode, which is then associated with disk blocks that hold the file's content. This integration with the VFS allowed file system calls to modify inodes and disk blocks through S5FS.
- **Virtual Memory (VM):** The final part involved implementing a comprehensive virtual memory system. VM enabled the kernel to manage user address spaces, execute user-level code, and handle system calls. This required creating virtual memory maps, handling page faults, and implementing anonymous and shadow objects. I also implemented file system calls, including the fork syscall, which is essential for process creation and execution within Weenix. Completing VM integrated the previous components into a fully functioning operating system!

2 Relevant Screenshots

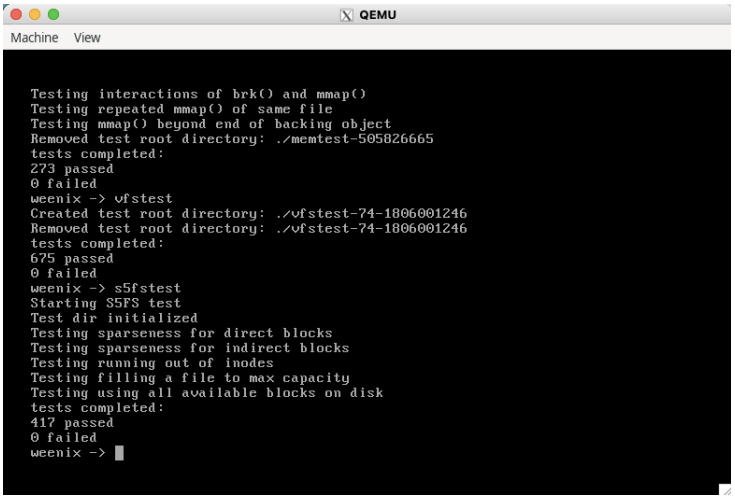
(a) Weenix after starting up



(b) Calling file system command (ls) and running a program (stat) which details underlying information about a file.



(c) Stress test for Weenix that tests various capabilities of complete OS. Culminates in a Forkbomb that ensures memory is not leaking



(d) Weenix runs over 1300 tests to ensure successful implementation of each component and their interactions.

(e) Following the above testing, Weenix halts cleanly which indicates proper management of references and successful cleanup of resources.