Operating System Feature Comparison: I/O and Provided Functionality

Andrew Chase

OS II

Spring 2015

Abstract: Compares and contrasts I/O devices and algorithms in Linux, Windows, and FreeBSD.

CONTENTS

I. File systems

Linux and FreeBSD both comes from a unix background. That means that the I/O layer is heavily interconnected with the filesystem. In Linux and FreeBSD pipes, sockets, and virtual memory can be mapped through the filesystem layer.

In Windows, these utilities are separated out in separate apis.

A. Filesystem Formats

Windows primarily uses the NTFS file system, but has built in support for CDFS, UDF, FAT12/16/32. Linux and uses the extensible file system, while FreeBSD uses the Unified File System, with ZFS available for use as well .

B. Locks

FreeBSD and Linux have the ability to lock parts of a file.

C. Data Structures

Windows stores files in a structure called the Master File Table. Each record in the table represents a file, and each record is at least 1k large. Files are represented by a file record number which is the index in the master file table. Each file name can be as long as 255 characters and can be unicode.

FreeSBD uses something called an index node, or inode. This contains much metadata about files (except filenames, which is contained in directories). There's also vnodes which are unique to a running process.

Linux ha a structure called the bio. This represents a block i/o structure.

D. Recovery

Windows maintains a journal. Every 5 seconds it writes a checkpoint, and during disasters it can save up to that checkpoint.

FreeBSD records a circular journal of metadata updates. It blocks operations if it fills. If a system experiences a hard fault, fsck runs through the journal to clean up the file system.

II. Block Systems

All the operating systems have ways of dealing with Block I/O devices, particularly magnetic ones since those used to be common in computers; and still are common for large volume storage. The priority is scheduling different requests so that no request gets ”starved” (never served) but also so the magnetic head spends as little time seeking as possible.

Linux uses the completely fair I/O elevator which merges and sorts to reduce seeks but provides some fairness among different requests processes by using queues and only pulling a certain number of requests per process. Also notable is the Noop elevator which merges requests, but does first-come-first-server other then that.