Part II: Contemporary software

FreeBSD is probably the most widely-deployed descendant within the Berkeley Software Distribution subtree of the UNIX family tree. Three others worth knowing about are NetBSD (whose intent is to run on as many different machine types as possible—their website currently claims support for 57 platforms), OpenBSD (whose intent is to ship the most secure operating system they can), and DragonflyBSD. In this part, you will explore the technical reasons for the existence of DragonflyBSD, and along the way hopefully learn a bit about the social side of open-source software development.

What was the primary point of contention that led to the creation of DragonflyBSD?

DragonflyBSD was created by Matthew Dillon in 2003 as a fork from FreeBSD 4.8. Dillon believed that there were issues with the way FreeBSD was choosing to implement threading and symmetric multiprocessing, and that over time these issues would lead to poor performance and maintenance problems. However, the FreeBSD project maintainers did not agree with these issues. So, after being barred from directly change the codebase, Dillon started the DragonflyBSD project.

wikipedia.org/wiki/DragonFly_BSD

dragonflybsd.org/history

osnews.com/story/6338/

What are the claimed benefits of the DragonflyBSD approach?

When DragonflyBSD was initially developed it was initially developed to support mulitcore processors. At the time only powerful high-end servers had clusters of processors. So, the operating system was created for these systems to avoid deadlocks as much as possible. More specifically it sought to make symmetric multiprocessing more straightforward, composable, understandable and algorithmically superior to the implementations in other operating systems.

Since then, the project moved toward optimizing the scalability of processors and reliability of the system.

dragonflybsd.org/performance/

makeuseof.com/what-is-dragonfly-bsd/

dragonflybsd.org/features/

phoronix.com/scan.php?page=article&item=corei9-freebsd13-dfly6&num=1

Give a very brief technical overview of the approach.

In place of the symmetric multiprocessing support (meaning a single processing done by different processors with the same operating system and memory) that was being brought forward in FreeBSD 5 branch, "DragonFlyBSD uses the concepts of partitioning and replication layered on top of a message

passing system to implement lock-free scalability on symmetric as well as non-symmetric NUMA multiprocessors."

The DragonFlyBSD operating system had a few technical goals that it wished to achieve in its inception. The first of which being the creation of lightweight threads to the FreeBSD kernel. The threads were lightweight in that user processes would have an associated thread and a process context, while kernels processes were "pure threads" with no process contexts. These lightweight threads would allow for high performance and simplicity for the scheduler.

The next goal was to create a message passing interphase between kernel threads and user threads: from one processor to another.

Finally the third technical goal of the project was in the implementation of multiprocessor support to allot resources to the specific processors. Then messages would be passed to preform operation on that resource. The goal was to create better synchronization techniques, and allow for greater scalability as the number of processors increase in a system.

A culmination in these technical changes ultimately allowed for each CPU has its own scheduler. When each thread was created it is assigned to a process and is never preemptively swapped out, ie, only switches when it blocks. The threads however can be migrated by the passing of an inter-processor interrupt (IPI) message between the CPUs involved. This consistency of processor allowed for better compartmentalization of a processes resources. Allowing for higher performance by allowing each processor in the machine the ability to use its own cache to store process data without waisted duplicate data in other processor caches.

wikipedia.org/wiki/DragonFly_BSD

people.freebsd.org/~hsu/publications/dragonflybsd.asiabsdcon04.pdf

---

How has this approach evolved over the years since its creation?

As mentioned above the project moved away from maximizing the efficiency of mulitcore processors, and instead focused toward optimizing the scalability of processors and reliability of the system.

Furthermore, since its inception it has also introduced its own filesystem HAMMER and later HAMMER2. This file system included a feature rich designed analogous to the increasingly popular ZFS filesystem.

academickids.com/encyclopedia/index.php/DragonflyBSD