MAXXCAT

Deciding between a hardware or software solution for Enterprise Search....

Implementers of enterprise search solutions have a fundamental decision to make between deploying a specialized piece of search hardware (MaxxCAT or Google), or a myriad of software solutions that range from toolkits and API's to full blown applications with complex administrative and configuration capabilities. This white paper outlines the considerable difference between the approaches in an effort to educate implementors on the relative merits of each approach.

Performance

If not the most important criteria, performance must rank highly in any candid evaluation. Software-only solutions are limited not only by the performance of the software itself, but also the underlying hardware, and the limitations imposed onto the software by the operating system. Increasing the performance of software-only solutions is not necessarily as easy as just running it on faster hardware. Doubling the clock speed, or number of cores in a general purpose server may have little to no impact on the performance of the software. Even techniques such as increasing physical memory, using faster disk drives, or multiple drives (which are more or less hardware techniques anyway) are hit or miss because the general purpose software running on general purpose hardware is not tuned, sized, or configured into an optimal configuration.

Furthermore, in order to run on the largest number of configurations, software only solutions must be built to the least common denominator of their projected installed base. Portability concerns, operating system dependencies and the inability to take advantage of specialized hardware including coprocessors or FPGAs hamper the design of general purpose software. There are many substantial differences between Operating Systems -- the Microsoft NTFS file system has been optimized for one set of criteria, where as with Linux, there a wide variety of file systems available, all optimized for different criteria. In order to run optimally on Solaris, NT, or any of the multitude of flavors of Linux, a software solution developer would have a combinatorial challenge to maintain a code base that worked optimally on each platform.

One of the many reasons that the MaxxCAT product line offers performance, in some cases multiple orders of magnitude faster than software solutions, is because the solution is built as a single monolithic component. MaxxCAT engineers do not have to write device interfaces for every file system out there, or accept the limitations of a general purpose file system that is the LCD of the installed base. Instead, a single file system, the best file system for our purposes, is selected, improved and optimized. And the layers of code above the file system can take advantage of the fact that the file system has been tuned to perform optimally for the single task of high performance search. Portability to other file systems, operating systems, or processors is not a design constraint in any way. Instead, the monolithic unit is built at each level to perform its single task as well as possible.

Even slower performing appliances like the Google Mini have the benefit of being built for a single, known hardware environment, and can attempt to make the best use of the available hardware. The distinction between MaxxCAT and Google becomes the question of, at a given dollar level of investment, who gets more performance from the hardware.

Simplicity

Even the simplest of software solutions require a certain amount of handholding to install and administer. Software presupposes a working hardware platform, but there is also administrative overhead in installing the software, configuring the software, backing up the software, upgrading the software etc. This is all in addition to the tasks required to keep the underlying hardware running.

Specialized hardware appliances, be they firewalls, mailservers, or in the case of Enterprise Search, dedicated search hardware, have the intrinsic advantage of operating as a "black box" in the case of Google, or as a "grey box" in the case of MaxxCAT. This box paradigm abstracts away much of the underlying complexity of the hardware, software and operating system, and exports a simple, but hopefully robust, API that can be manipulated to achieve the desired results with a minimum of time spent tuning, configuring, patching, installing etc. MaxxCAT search appliances can be up and performing their job in as little as 5 minutes. Plug them in, hook up the network and search.

Well designed software can have the perception of being simpler to customize -- it's software after all. Just change some bits around and go, whereas with hardware, it involves moving physical devices, installing things, wiring things etc. So software has a potential edge here. However, most of the things that affect enterprise search at the lower levels of the problem do not need to be configured. Register word sizes are fine at 64 bits, the amount of disk storage can be safely fixed at some upper bound and so on. Customization comes in only at the top level of the solution -- the user interface. What does the user see, what data sources does the appliance connect to and so on.

A well designed search appliance, such as the MaxxCAT EX5000 provides maximum configurability and customization at the top level of the application stack, by exporting an extremely simple mechanism for retrieval. The hardware can rapidly locate relevant results, interacting optimally with internal subsystems with zero configuration by the user, but the part that matters, the part that needs to be customized is left exposed at the top level. So in this sense, hardware gives all of the simplicity of customization as software at the top level, while obscuring all of the nitty-gritty details of buffer allocation, disk scheduling, interrupt handling etc from even the need to be customized or tuned. Not so with software -- the software has to be installed in a certain location, it has to be registered with the operating system for start up at boot time, it has to have security access to the required ports, etc. And then, the software still needs to be customized. And since software is so "easy" to customize, developers often make it much more complicated, burdening the integrator with too many choices, too many parameters, too many levels of detail. So the simplicity is lost, even to the software.

Other costs creep in when simplicity is lost. With a hardware solution, either the appliance is working, or it is not. However, with software, tracking down problems is much more time consuming. The software vendor blames the hardware vendor (or the customer!). The hardware vendor blames the operating system. The OS vendor blames the external storage manufacturer. And the customer is in the middle, playing traffic cop. With a search appliance, it is obvious where the problem is, and much simpler to resolve. At least in the case of MaxxCAT, hardware problems can be solved with a single phone call to one vendor, and possibly the replacement of faulty hardware.

MAXXCAT

Deciding between a hardware or software solution for Enterprise Search....

Scalability

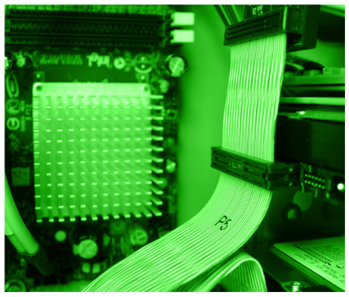
Another area where there is a big distinction between hardware appliance based search solutions and software solutions is how scaling is handled. Scalability is where performance meets simplicity and either things get much more complicated, or they don't. In the case of a software solution, moving from 1 node to 2 nodes involves first of all, acquiring, configuring possibly testing more hardware, or finding a suitable device from existing inventory. And then the same old process begins anew -- install the software on the second machine, patch the software on the second machine, configure the operating system on the second machine etc. Then, depending upon the software, the data and user customizations may have to be migrated to the second node, applications that use the software may have to be reconfigured to become aware of the additional node, API config files may have to be adjusted etc. Not very simple.

Within the MaxxCAT product line, adding an additional node is as simple as acquiring the second device from MaxxCAT, plugging it in and connecting it to the network. Because the MaxxCAT design is monolithic, even at the cluster level, the nodes are self-aware, self activating and can easily synchronize data, redistribute the query load, and require no additional configuration or modification of user interfaces. This simplicity is only possible because all of the components are designed to work together, and therefore, much of the complexity of converting a single instance of a search solution into a parallel instance is done inside the box, not by an administrator or technical expert.

Finally, once applications move from single threaded, single instance, to multi-homed, things become much more complicated. Issues of concurrency, synchronization, latency are now foremost, and although these features are certainly programmed into software solutions, they suffer from the same limitations as general purpose hardware running general purpose software. The synchronization mechanisms, the latency time outs, etc have to drawn from the pool of LCDs, and have to either work across the universe of possible installed base, or allow for the user, again, to configure parameters to tune the distributive aspects of the solution. But this is not the case for well designed, monolithic hardware solutions like MaxxCAT, where the configuration and optimization are built into the product, not left for the customer to struggle with.

Life Cycle

One argument that could be constructed in favor of software is that the development of specialized hardware can not keep pace with the development of software, that is to say that hardware, thanks to Moore's Law, becomes obsolete very quickly, every 18 months perhaps. It is but a small matter, the argument goes, for software to take advantage of any new developments in general purpose hardware simply by upgrading the hardware underneath the software, and presto, the software-only solution is now 18 months ahead of the special purpose hardware solution. Also contributing to this reasoning is that the speed of general purpose hardware increases much faster than the speed of specialized hardware because the cost and pace of development of general purpose hardware is spread across a much wider installed base. As such, it is only a matter of time for tomorrow's general purpose laptop to be faster than today's special purpose super computer.



But a hardware-only solution can obviate this objection by simply building its hardware platform from generally available commodity hardware, as opposed to specialized board or circuit designs. Indeed this is the approach MaxxCAT has taken. By designing units from commodity hardware, Moore's Law works in favor of hardware solutions, not against it. Just as software can become faster by virtue of the performance gains in the underlying general purpose hardware, so can hardware appliances. In either case, the hardware must be upgraded to take advantage of newer technology, however, the upgrade path on a hardware-only solution is much simpler. Just upgrade the hardware (plug it in, hook it to the network). With software though, the underlying general purpose hardware, operating system, network must be installed, and then the software must be loaded, and perhaps reconfigured to take advantage of new capabilities in the underlying platform.

In conclusion, we hope that this paper has presented some of the distinctions between hardware-based solutions and software-based solutions, and will help to persuade those who are not yet convinced to further explore the performance, simplicity and scalability aspects of specialized hardware solutions.