Why Auto-Tiering is Critical

By: Jim O'Reilly

Storage in IT comes in multiple flavors. We have super-fast NVDIMMs, fast and slow SSDs and snail-paced hard drives. Add in the complexities of networking versus local connection and price, and capacity, and figuring the optimum configuration is no fun. Economics and performance goals guarantee that any enterprise configuration will be a hybrid of several storage types.

Enter auto-tiering. This is a deceptively simple concept. Auto-tiering moves data back and forth between the layers of storage, running in the background. This should keep the hottest data on the most accessible tier of storage, while relegating old, cold data to the most distant layer of storage.

A simplistic approach isn't quite good enough, unfortunately. Computers think in microseconds, while job queues often have a daily or weekly cycle. Data that the computer thinks is cold may suddenly get hotter than Hades when that job hits the system. Similarly, admins know that certain files are created, stored and never seen again.

This layer of user knowledge is handled by incorporating a policy engine into auto-tiering, allowing an admin to anticipate data needs and promote data through the tiers in advance of need.

Auto-tiering has been around a while, usually in appliances that can relegate primary storage to secondary storage in the background. At best, these are partial solutions, with most just covering storage in the appliance and downstream from it, though we are starting to see more powerful holistic capabilities from companies such as Enmotus. Even though hard drives are obsolescent, the evolution of solid-state storage is putting much more emphasis on server-based tiering solutions, with hyper-converged systems and NVDIMMs changing the definition of primary tiers and storage networking. In fact, SSD is adding more tiers to the storage structure, increasing the need for tiering controls.

The ideal auto-tiering solution has to encompass all of these tiers. It should manage data in hyper-converged systems where all storage local to a server is shared over a network fabric. It has to handle traditional networked storage as well, in all of its flavors. This requires a solution deeply embedded in the data flow with device handlers that can reach across all of the device types. Currently, such a broad scope is seen only in the Enmotus solution, but the clear benefits the approach brings should cause "omni-tiering" to be ubiquitous in server stacks and hyper-converged systems over time.

Inside the server, data has to flow from DRAM to NVDIMM to NVMe drives and then on to slower storage. This is both a bigger technical challenge and the largest source of cost/performance benefit in next-generation servers.

These computers will use large, tightly coupled DRAM spaces with capacities in the low terabytes and will be ideal as shared-memory database systems, for example. Couple an NVDIMM persistent storage

layer with maybe 10 terabytes of fast solid-state memory and that's quite a storage pool already. Outside of this closely-coupled memory complex, we'll have a set of M2-sized (that's really small!) NVMe SSDs with capacities from 2 to 10 terabytes each. All of this is going to be shared, most likely over Ethernet links using RDMA.

Not all servers will be in this high-performance range, of course. There is still a need for servers that run low to mid-range virtual machine pools. These servers will still have a tiered memory system. With local SSD for instance storage and large pools of networked storage a secondary tier. If an auto-tiering solution is well designed it can cope with this architecture too, which really simplifies management of the whole IT shop.

Big Data poses challenges with auto-tiering solutions. Creating huge files, it would seem to make any tiering activity very inefficient. This brings up the issue of tiering blocks versus objects. If part of a huge file is hot and the rest cold, promoting just the hot segment is very beneficial, while avoiding the high traffic associated with moving huge objects and then storing them.

Block-level auto-tiering seems to be the correct approach for most storage pools. It can handle all of the active data, while traditional backup and archive approaches can take objects as entities and move them to cold storage such as the public cloud archiving spaces.

With so much churn in performance and in the way apps will be organized and run, auto-tiering packages need to have analytics capabilities built in. From knowing where data is to monitoring job performance peak loads to spotting traffic bottlenecks by individual storage node or device, a good monitor can make it easy to exceed customer expectations for service.

It goes without saying that good software is strongly fault-resistant. The death of a device will happen and this has to be handled to prevent data loss ensure fast access to alternate copies of any bad block. Having striping and mirroring capabilities is thus important, bringing opportunities for parallel access if needed.

High-complexity storage needs automated tools and auto-tiering is one that will ease an admin's work by managing data positioning optimally and tracking system bottlenecks and failures.

About the author

Jim O'Reilly is a well-respected consultant and commentator on the IT industry. He regularly writes for top-tier publishing sites. Jim is also an experienced CEO with a track record that includes leading the development of SCSI and built the industry's first SCSI chip, now in the Smithsonian. Jim's profile is at

https://www.linkedin.com/in/jamesmoreilly