Shashi Shivaraju
XID:C88650674

Paper Review 4 - Memory Resource Management in VMware ESX Server

VMware ESX Server is a thin software layer designed to multiplex hardware resources efficiently among virtual machine monitors [VMM] running unmodified commodity operating systems. ESX Server manage system hardware directly whereas VMM needs a hosted OS thus providing significantly higher I/O performance and complete control over resource management when compared to VMM. VMware ESX Server requires an efficient memory management techniques to allocate memory across virtual machines running existing operating systems without modification. This paper describes several novel mechanisms and policies for managing memory in VMware ESX Server.

A guest operating system that executes within a virtual machine expects a zero-based physical address space, as provided by real hardware. ESX Server satisfies this by providing each VM with memory virtualization by adding an extra level of address translation. To achieve this ESX Server provides Pmap for physical-to-machine page mapping and Shadow page table for virtual-to-machine page mapping. This method has no additional performance overhead since hardware TLB will cache direct virtual-to-machine address translations read from the shadow page table. It is flexible since ESX Server can remap a “physical page” by changing pmap and can monitor guest memory access.

ESX Server supports memory over-commitment,wherein the total size configured for all running VM exceeds the total amount of actual machine memory. When memory is over-committed, ESX Server should reclaim space from one or more of the VMs. Conventional page replacement introduced by an extra level of paging has poor performance and many issues,so the authors of the paper propose a technique known as Ballooning,in which ESX Server implicitly coaxes a guest OS into reclaiming memory using its own native page replacement algorithms. In Ballooning, the VM from which memory has been reclaimed performs as if it had been configured with less memory. Since ballooning totally depends on guest OS, ESX Server has limited control if the balloon drivers are uninstalled or disabled explicitly or if the guest OS is booting. Compared to conventional page replacement the performance of ballooning is faster.When ballooning is not possible or insufficient, the system falls back to a paging mechanism. Memory is reclaimed by paging out to an ESX Server swap area on disk, without any guest involvement.

ESX Server supports sharing memory between virtual machines. Conventional transparent page sharing introduced by Disco requires guest OS modifications and thus cannot be used by ESX Server. ESX_Server supports Content-based Page Sharing wherein page copies are identified by their contents and pages with identical contents can be shared. Hashing is used to identify common pages, wherein a hash function computes a checksum of a page, which is used as a lookup key and chaining is used to handle collisions. Comparing each page with every other page is computationally heavy O(n*n),so in implementation it is done randomly.

ESX Server provides quality-of-service guarantees to clients of varying importance .To achieve this ESX Server employs a new allocation algorithm based on share-based allocation with Idle Memory Taxation. Here a client is taxed more for an idle page than for one it is actively using. When memory is scarce, pages will be reclaimed preferentially from clients that are not actively using their full allocations. A tax rate specifies the maximum fraction of idle pages that may be reclaimed from a client. ESX Server computes a target memory allocation for each VM based on both its share-based entitlement and an estimate of its working set using the algorithm. ESX Server provided three basic parameters to control the allocation of memory to each VM: a min size, a max size, and memory

shares. The min size is a guaranteed lower bound on the amount of memory that will be allocated to the VM, even when memory is over-committed. The max size is the amount of “physical” memory configured for use by the guest OS running in the VM. Unless memory is over committed, VMs will be allocated their max size. Memory shares entitle a VM to a fraction of physical memory, based on a proportional-share allocation policy.

ESX Server provides admission control which ensures that sufficient unreserved memory and server swap space is available before a VM is allowed to power on. Machine memory must be reserved for : min size + overheads and the disk swap space must be reserved for max size – min size. These reservations ensures the system is able to preserve VM memory under any circumstances.

ESX Server also provides Dynamic Reallocation wherein memory allocations are recomputed in response to changes to system-wide or per-VM allocation parameters,addition or removal of a VM to/from the system,changes in the amount of free memory that cross predefined thresholds or changes in idle memory estimates for each VM. Four thresholds are defined to reflect different reclamation states and they are 1) High (6% of system memory) wherein no reclamation is performed 2) Soft (4% of system memory) wherein reclamation is performed through ballooning or paging 3)Hard (2% of system memory) wherein reclamation is performed through paging and 4) Low (1% of system memory) wherein reclamation is performed through paging paging with execution of all VMs blocked.In all states, system computes target allocations for VMs to drive the aggregate amount of free space above the high threshold and the system transitions back to the next higher state only after significantly exceeding the higher threshold to prevent rapid state fluctuations.

To test their work, the authors of the paper perform many benchmark tests to measure memory metrics. Experiments were carefully designed and tested on ideal scenarios and real-life workloads. The results back the ideas proposed in the paper and showed that the overhead was nominal when compare to the advantages the proposal provides. Thus the authors argue that their proposed methods provides efficient memory management in VMware ESX Server.