Paper title: Memory Resource Management in VMware ESX Server
Paper authors: Carl A. Waldspurger
Your name: Parth Kansara (115135130)

What problem does the paper address? How does it relate to and improve upon previous work in its domain?
The paper talks about the growing popularity of server virtualization and the utilization of virtual machines for the same. These VMs allow different OSs to run parallelly and share the underlying hardware resources through a single interface. However, none of the existing solutions are able to perform efficiently without tweaking the operating systems. Further, the existing policies for reclamation, page sharing, allocation etc. have several drawbacks and performance bottlenecks that need to be overcome to efficiently support multiple OSs.

IBM's VMM and **Disco** are some examples of previous VMMs. These products required several modifications to the guest operating system before running it on the VMs, unlike the ESX Server presented in the paper. Unlike the previous product, **VMware Workstation**, which used a hosted architecture for portability, the ESX Server provides management of the server hardware directly. The self-paging technique used in the **Nemesis system** for memory reclamation requires application to manage the virtual memory operations, which is counter-productive due to the existence of applications that do not implicitly have that ability. EXS Server provides an alternative technique that is more efficient. The existing transparent page sharing technique used in **Disco** influenced the ESX Server's content-based page sharing which avoids any modifications to the guest OS and provides more opportunities for sharing. Previously utilized page-fault mechanism for working set estimation is enhanced by statistical sampling in the ESX Server. The existing algorithm on proportional-share allocation is improved by the introduction of the idle tax, which performs better memory utilization.

What are the key contributions of the paper?
The paper addresses the above issues and presents the VMware ESX Server, which is a software interface for allowing VMs with different operating systems to interact with the hardware resources on the host system. It also demonstrates several original techniques related to memory resource management. To provide an operating system with a zero-based physical address space, the ESX Server uses the *pmap* data structure to translate the physical page numbers to machine page numbers for each VM. Further, *shadow page tables* are maintained for the processor, which contain virtual to machine page mappings. In cases of overcommitment of memory, the ESX Server performs memory reclamation using the ballooning technique. It loads a balloon module into the guest OS which can inflate, increase memory pressure on the guest OS, and get allocated pages by the guest OS which performs memory reclamation as it sees fit. These allocated pages, which are now free, are communicated to the ESX Server, which can free them without disrupting the VM. When a physical page is allocated to the balloon module, the ESX Server marks its pmap entry and deallocates the corresponding machine page. This makes sure that the guest OS doesn't try to take those pages back, as the missing pmap entry will generate a fault. In cases where ballooning isn't enough, the system switches to a randomized page replacement mechanism.

The ESX Server allows sharing of pages with identical contents, which avoids any modification of the guest OS, while allowing increased sharing. The page contents are summarized using a hashing function, and is checked against a hash table to find a matching page. In case of a match, an in depth comparison is carried out to verify that the pages are identical. If matching, the pages are shared until a write operation for the page occurs, which results in the creation of a private copy for the writer. If a page doesn't have any match, it is tagged as a special *hint* entry instead of storing it in the hash table to avoid any write overheads. For memory allocation, the ESX Server modifies the existing proportional-share based algorithm to include an idle memory tax. Thus, if clients do not fully use the allocated memory, then the idle memory is revoked. The ESX Server defaults to allowing 75% of the idle memory to be revoked, but it can be changed. The idle memory is measured by sampling each VM and tracking a small number of uniformly distributed physical pages. The memory allocation process in the ESX Server is dynamic to adapt to the various events in the system, along with periodic rebalancing based on the idle memory of each VM.

Briefly describe how the paper's experimental methodology supports the paper's conclusions.
The techniques used in the ESX Server have significant impact. The content-based page sharing algorithm demonstrates up to 67% sharing as the number of VMs increases, while requiring a negligible CPU overhead. Due to the idle memory tax incorporated in the proportional share-based memory allocation technique, the ESX Server was able to increase the throughput of the workload by 30%. Based on all these techniques along with dynamic memory reallocation, the ESX Server was able to provide a total memory of 1200 MB across all VMs, while the host machine memory was only 1GB. These results prove that the ESX Server can efficiently allocate memory resources for VMs running diverse OSs.
Write down one question you may want to ask the authors.
What could be future optimizations that can aid an easy and mass adoption of the VMware ESX Server?