Memory Overcommitment In ESX

Goals

- When demand of all the VMs can be met, they should be met
- When the resources are less than demand, they should be distributed amongst the VMs according to their importance.

Techniques Used

- Page Sharing - Share pages which have the same content across VMs. Very useful since VMs may be running similar OS and applications.
- Memory Ballooning - Reclaim free memory from VMs
- Page compression - Compress pages with a compression ratio of 2 or more
- Hypervisor level swapping
- Block page allocation by VM

Sharing, Ballooning and Compression are opportunistic techniques and do not guarantee reclamation, while swapping does.

Memory States of ESX

- **High** - 6% free memory in ESX. Uses page sharing, i.e. page sharing is active at all times
- **Soft** - 4% free memory. Page sharing and Ballooning. If a VM has consumed pages and is not using it, balloon driver reclaims the memory.
- **Hard** - 2% free memory. Sharing, compression and swapping. ESX aggressively reclaims memory by swapping. If the page is shareable or compressible, do that instead of swapping.
- **Low** - 1% free memory. sharing, compression, swapping and blocking. This is like hard state, but ESX also prevents VMs from allocating more memory till ESX reaches hard state by memory reclamation.

Memory States of ESX

- The 4% buffer is called `minfree`. ESX is in comfortable state as long as free memory is more than `minfree`
- Page Sharing happens in all the memory states
- Ballooning starts at 4%.
- If ballooning is not able to reclaim memory and take ESX back to `high` state, it enters the `hard` state where it starts swapping, and subseqently low state where page allocations are blocked.
- After reclamation, system transitions to a higher state only after significantly exceeding its threshold. This is to prevent rapid oscillation between states.

Memory Classification

- **Mapped Memory** - All memory pages of a VM which are ever accessed by a VM are considered mapped by ESX. Can be more than total ESX memory in case of overcommitment. A mapped page may be either backed by a physical page or be shared/compresses/ballooned/swapped(by ESX or guest OS).
- **Consumed Memory** - The memory of VM which is backed by a physical page. Total consumed memory of all VMs has to be strictly less than total ESX memory.
- **Working Set/Active Memory** - If total working set of all VMs is more than ESX memory, VMs will thrash and memory reclamation techniques will not not useful.

Conflict

How much memory to reclaim from **which** VM?

Resource Sharing

- Each VM is allocated some number of shares for memory. A machine is guaranteed a minimum resource fraction equal to its fraction of the total shares in the system.
- Naive technique -
  - In case of overload, memory is revoked from client that has **fewest shares per allocated page**.
  - Drawback - Idle client with lots of share can hoard memory unproductively while active clients with fewer shares are under memory pressure.

Idle Memory Tax

- Imposes a tax on idle-memory.
- Tax rate specifies the maximum fraction of idle pages that can be reclaimed from a VM. In ESX, its default value is 75%
- How to find idle memory? Cannot rely on OS metrics(they rely on access bit in page tables which are bypassed by DMA for deice I/O) Samples memory of VM to find out fraction of pages accessed in a time interval.
- Can this be done in a better way? Memory is cheap, there is no need to calculate idle pages, just allocate the free pages

Cost of different techniques

- **Page sharing** - small memory cost to store metadata, negligible CPU cost to identify pages that can be shared, copy on write cost when VM tries to write to shared page
- **Ballooning** - small cpu cost to identify pages that can be reclaimed, cpu cost on page fault when the ballooned page is released, and if balloon earlier resulted in guest swapping, then huge wait times for swapping them back in
- **Swap** - Huge wait times if the swapped pages are required by the VM

Comparison with KVM

- KVM has **memory ballooning, page sharing and hypervisor swapping**. What it lacks is a policy governing all these three features.
- A policy can be implemented in the hypervisor, or even outside it as a set of scripts.
- Need to investigate more on swapping. How to trigger swapping of pages?
- Possible policies -
  - Have a threshold like `minfree` and start ballooning after that. How much memory to take from each VM? Can be done on the basis of shares. Default can be equal shares to all.
  - Start swapping if not able to reclaim memory through ballooning. But swapping is controlled by the host kernel in KVM's case
  - When should a VM be migrated? How about when working set of a VM > hypervisor memory? or when consumed memory is > hypervisor memory?

References

- https://labs.vmware.com/vmtj/memory-overcommitment-in-the-esx-server
- http://www.vmware.com/pdf/usenix_resource_mgmt.pdf