Improved Memory Overcommitment handling in vSphere 6.0

Before I jump onto this topic, I want to highlight about Large Pages in VMware vSphere ESXi hosts.

Setting the stage

VMware ESXi provides 2MB memory pages, commonly referred to as “large pages” along with usual 4KB memory pages. ESXi will always try to allocate 2M pages for main memory and only on failure try for a 4K or small page. Virtual machines are large pages if 2M sequences of contiguous Memory Page Numbers are available. The idea is to reduce amount of page sharing and also increase the memory footprint of the virtual machines. The biggest benefit is of mitigating TLB-miss (Translation Lookaside Buffer) costs as much as possible for Nested Page Table enabled servers running ESX.

However, allocating memory in 2M chunk may cause the memory allocated to the VM to become fragmented. But as small pages are allocated by a guest and VM, these larger sequences need to be broken up.

So, if defragmentation occurs there could be enough memory to satisfy a large page request even when there is no 2M contiguous Memory Page Number’s available. The defragmenter’s job is to remap the existing allocated small pages in a 2M region to allow that range to be mapped with a large page.

Transparent Page Sharing runs across a host’s VMs periodically (every 60 minutes by default) and reclaims identical 4KB pages. However, this only happens when there are no large pages (2MB pages). If you use large pages (2MB), TPS does not come into picture because of the “cost” of comparing these much larger pages. That is until there is memory contention, at which point the large pages are broken down into 4KB blocks and identical pages are shared.

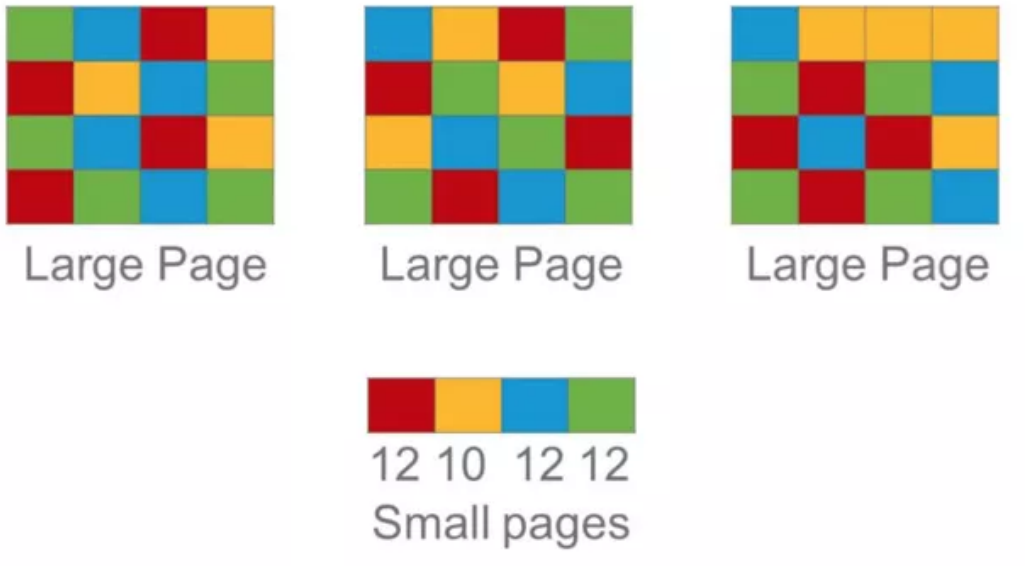
Large pages offer performance improvements. Reclamation of large pages does not happen if there is not enough physical memory to back all of the VMs memory requests.

Memory savings happens due to TPS kick in when you run lots of VMs with very similar memory usage on a host. You will not see any memory savings until your host thinks it is under pressure which will happen if the large pages don’t get shared. The host will wait
until it hits 94 percent memory usage (6 percent free), before it deems itself under memory contention and starts to break those large pages into smaller 4KB ones. In a typical environment where you have many similar VMs, you are consistently going to run your host at around 94 percent memory used. All those identical memory pages can still be reclaimed, just as before, and you are gaining a performance gain of large pages.

Improvement

ESXi backs guest physical memory pages with large host physical pages which are 2MB of continuous memory region instead of the 4KB for regular pages for better performance. ESXi will not share large physical pages because the probability of finding two pages that are identical is very low.

When the host is overcommitted there are three main mechanisms vSphere uses to reclaim memory in this order: Transparent page sharing, ballooning and swapping. Before any of these can occur. Large pages need to be broken into small pages. This takes a bit of time.



Before 6.0, ESXi hosts didn’t leave much time for the large page break down and page sharing to happen before ballooning kicked in, which had a performance impact on the host as ballooning is more expensive than transparent page sharing.

In vSphere 6.0, ESXi hosts allow more time for the large page break down and page sharing to happen before ballooning kicks in, improving the overall handling of memory overcommitment.