Another way - level of indirection

Three abstractions of memory

- Machine - actual hardware memory (16 GB of DRAM)
- Physical - abstraction of hardware memory managed by OS
  If a VMM allocates 512 MB to a VM, the OS thinks the computer has 512 MB of contiguous physical memory (underlying machine memory may be discontiguous)
- Virtual - virtual address spaces (similar to virtual memory)
  standard 2^{32} or 2^{64} address space

➡ Translation - from **VM's Guest VA** to **VM's Guest PA** to **Host PA**
in each VM, OS creates and manages page tables for its virtual address spaces without modification but these page tables are not used by the MMU hardware

Shadow page tables

VMM creates and manages page tables that map virtual pages directly to machine page these tables are loaded into the MMU on a context switch

➡ **VMM page tables are the shadow page tables**

VMM needs to keep its virtual to machine tables consistent with changes made by OS to its virtual to physical tables

- VMM maps OS page tables as read-only (i.e., write-protected)
- When OS writes to page tables, trap to VMM
- Memory tracing :VMM applies write to shadow table and OS table and returns
- Memory-mapped devices must be protected for both read- and write- protected