One way - direct mapping

➡ VMM uses the page tables that a guest OS creates (direct mapping by MMU)

VMM validates all updates to page tables by guest OS

- OS can read page tables without modification
- but VMM needs to check all page table entry writes to ensure that the virtual-to-physical mapping is valid

◉ Requires to modify the OS to patch updated to the page table (used in *Xen* paravirtualization)

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