Virtualizing memory

OS assumes to be in full control over memory via the page table

But VMM partitions memory among VMs

- VMM needs to assign hardware pages to VMs
- VMM needs to control mappings for isolation
  - Cannot allow an OS to map a virtual page to any hardware page
  - OS can only map to a hardware page given to it by the VMM

Hardware-managed TLBs make this difficult

- When the TLB misses, the hardware automatically walks the page tables in memory
- As a result, VMM needs to control access by OS to page tables

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)