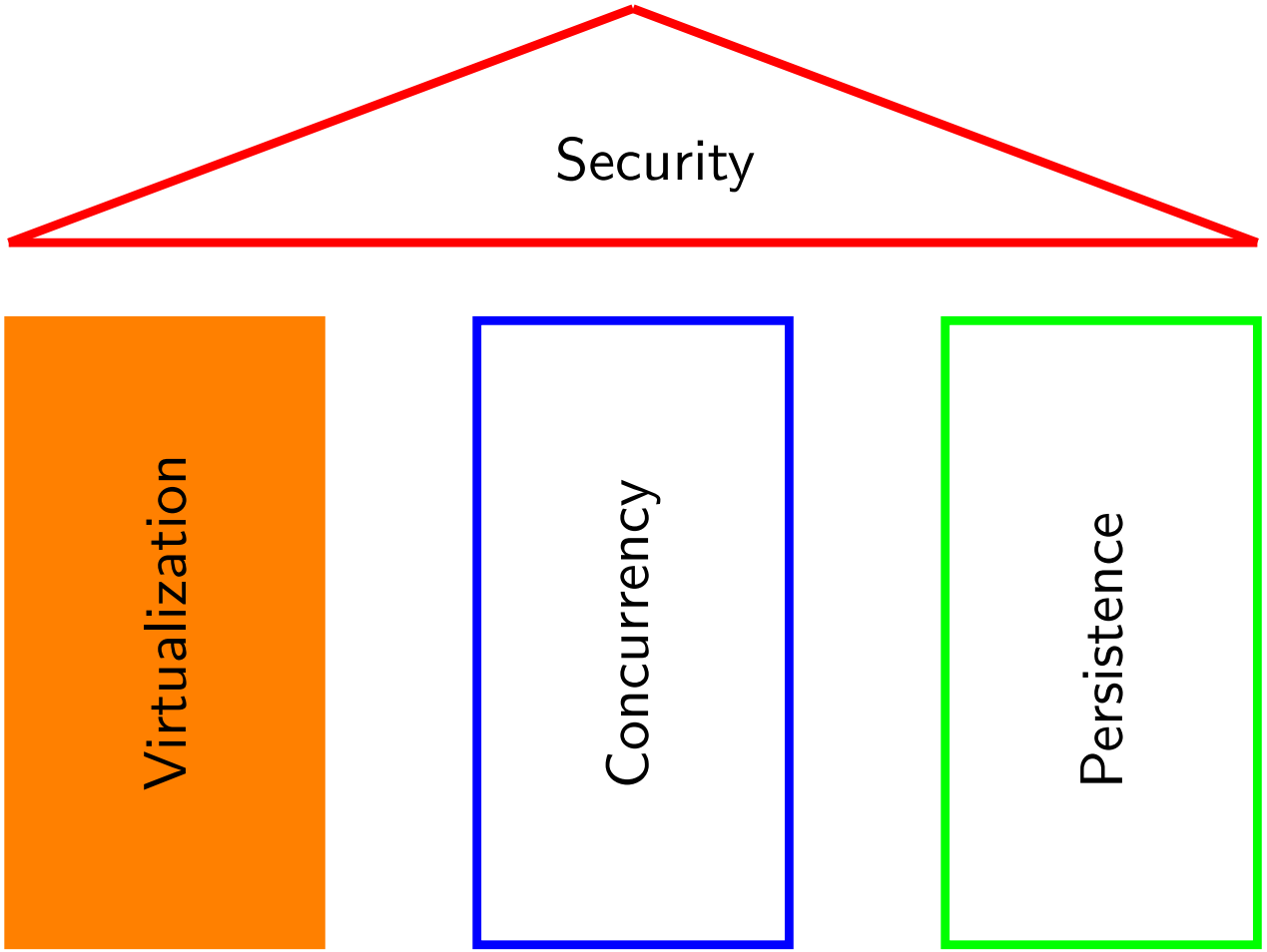
CS323 Operating Systems

Virtualization Summary

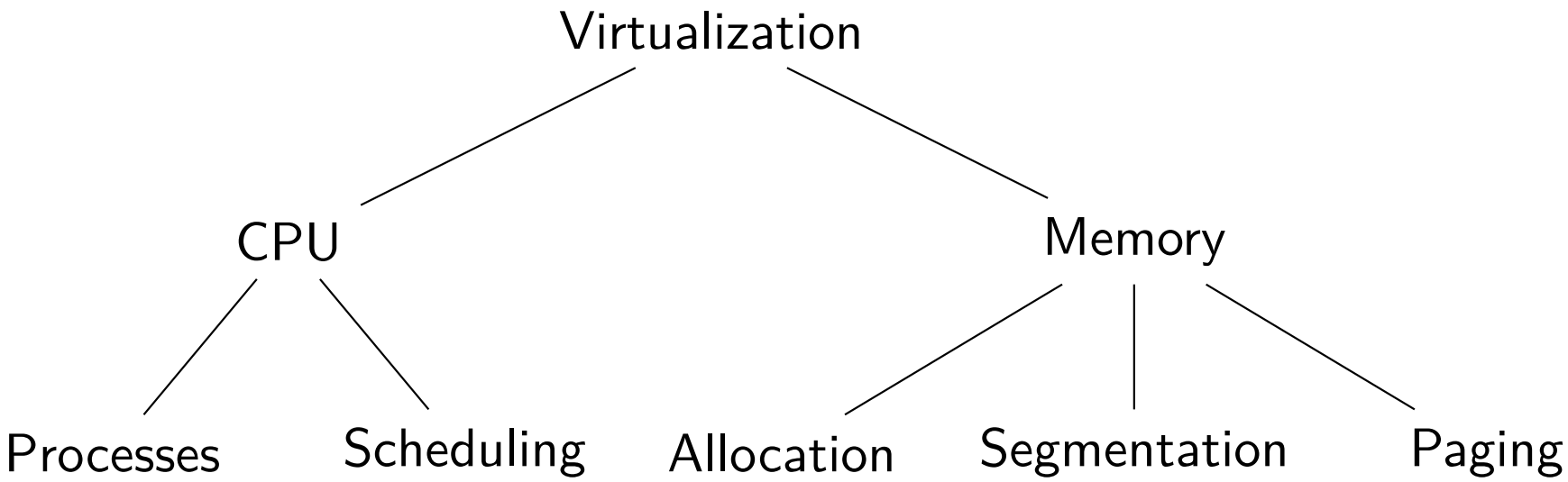
Mathias Payer and Sanidhya Kashyap

EPFL, Fall 2021

Virtualization



Virtualization: Summary



CPU Virtualization: Processes

- Processes are a purely virtual concept
- Separating policies and mechanisms enables modularity
- OS is a server, reacts to requests from hardware and processes
- Processes are isolated from the OS/other processes
  - Processes have no direct hardware access
  - Processes run in virtual memory
  - OS provides functionality through system calls
- A process consists of an address space, associated kernel state (e.g., open files, network channels) and one or more threads of execution

CPU Virtualization: Scheduling

- Context switch and preemption are fundamental mechanisms that allow the OS to remain in control and to implement higher level scheduling policies.
- Schedulers need to optimize for different metrics: utilization, turnaround, response time, fairness and forward progress
  - FIFO: simple, non-preemptive scheduler
  - SJF: non-preemptive, prevents process jams
  - STFC: preemptive, prevents jams of late processes
  - RR: preemptive, great response time, bad turnaround
  - MLFQ: preemptive, most realistic
  - CFS: fair scheduler by virtualizing time
- Past behavior is good predictor for future behavior

Memory Virtualization: Segmentation

- OS manages access to constrained resources
  - Principle: limited direct execution (bare metal when possible, intercept when needed)
  - CPU: time sharing between processes (low switching cost)
  - Memory: space sharing (disk I/O is slow, so time sharing is expensive)
- Programs use dynamic data
  - Stack: program invocation frames
  - Heap: unordered data, managed by user-space library (allocator)
- Time sharing: one process uses all of memory
- Base register: share space, calculate process address through offset
- Base + bounds: share space, limit process' address space
- Segments: movable segments, virtual offsets to segment base

Memory Virtualization: Paging

- Fragmentation: space lost due to internal or external padding
- Paging: MMU fully translates between virtual and physical addresses
  - One flat page table (array)
  - Multi-level page table
  - Pros? Cons? What are size requirements?
- Paging and swapping allows process to execute with only the working set resident in memory, remaining pages can be stored on disk

Book chapters

- Virtual CPU (Processes and Threads): OSTEP 4–6
- Virtual CPU (Scheduling): OSTEP 7–10
- Virtual Memory (Segmentation): OSTEP 13–17
- Virtual Memory (Paging and Swapping): OSTEP 18–22

This concludes the first pillar of OS.