The textbooks authors repeatedly mention that address translation is a power tool and enables multiple policies. The overall framework is given in Figure 8.1 (slide 3 of Chapter 8A). For each of the following, identify the address translation mechanisms used and briefly explain how they are used.

1. Process isolation – “Keep a process from touching anyone else’s memory, or the kernel’s”. See sections 2.2.2 and 8.2 (slides 7, 11, and 20 of Chapter 8A).

Relocation registers, Segment Table, or Page Table enforce Process Isolation.

2. Shared code segments – “E.g., common libraries used by many different program”. See Figure 8.4 (slide 15 of Chapter 8A).

Segmentation – Have each process points the segment table entry to the same (execute-only) code segment => reentrant code (no changes to code segment between process switches).

Paging – If the code segment starts on a page boundary and ends up on a page boundary (even with some internal fragmentation in the last page) => reentrant code (no pages with both shared code and process-specific data).

3. Demand-paged virtual memory – “Illusion of near-infinite memory, backed by disk”. See section 9.6.2 (slides 10, 11, and 12 of Chapter 9).

Page tables start with the presence bit off. A reference to the missing page causes a page fault exception, which tells the kernel to bring in the missing page and then resume thread execution.

PTE contains presence bit, access permission bits, use bit, modified bit, and page frame (and more depending on processor).

4. Checkpoint/restart – “Transparently save a copy of a process, without stopping the program while the save happens”. See section 10.3.1 (slide 12 of Chapter 10).

Make a list of pages to copy to disk, turn all their access permissions to read-only, then restart the process execution. If there is a write permission exception to what should normally be a R/W page, we clone the page with original copy used by the checkpointing process and the cloned page is set to R/W permission and placed in the user process page table. This is essentially a copy-on-write.