Concepts of Programming Languages, CSCI 305, Fall 2021
Address Space, Section 15.3, pages 790-801, Dec. 1

Object files are typically divided into multiple sections which include the following sections:

- Import, export and relocation tables (for relocatable object files), along with the amount of space required by the program for noninitialized static data.
- Code
- Read-only data (constants and jump tables for case statements, etc.)
- Initialized but writable static data
- High-level symbol table

The first and last sections are not usually brought into memory at run-time.

- Import, export and relocation tables, along with the amount of space required by the program for noninitialized static data section is used by the linker and loader
- High-level symbol table is used by debuggers and performance profilers

There are two formats of object code files:

Relocatable	Executable
Contains objects that need to be updated, based on where items are placed into memory	Can be brought into memory and run without modification
Contains: • Import table • Export table • Relocation table	Contains: • No reference to external symbols • Defines a starting address for execution
Names in the import and export tables are referred to as "external symbols"	

Import table – Identifies instructions that refer to named locations whose addresses are unknown but are presumed to lie in other files yet to be linked to this one.

Export table – Lists names and addresses of locations in the current file that may be referred to in other files.

Relocation table – Identifies instructions that refer to locations within the current file, but that must be modified at link time to reflect the offset of the current file within the final, executable program.

Typical running program:

- Code
- Read-only data (constants and jump tables for case statements, etc.)
- Initialized but writable static data
- Uninitialized data – usually zero-filled.
- Stack – Usually given a small initial size, and extended automatically by the OS in response to (faulting) accesses beyond the current segment end
- Heap - Usually given a small initial size, and extended automatically by the OS in response to explicit requests (via system call) from heap-management library routines.
- Dynamic libraries – Modern OS typically arrange for most programs to share a single copy of the code for popular libraries.

  Tend to have a pair of segments:

  - Shared code
  - Linkage information and private copy of writable data library needs

Layout of process address space in x86 Linux, Figure 15.8, page 793

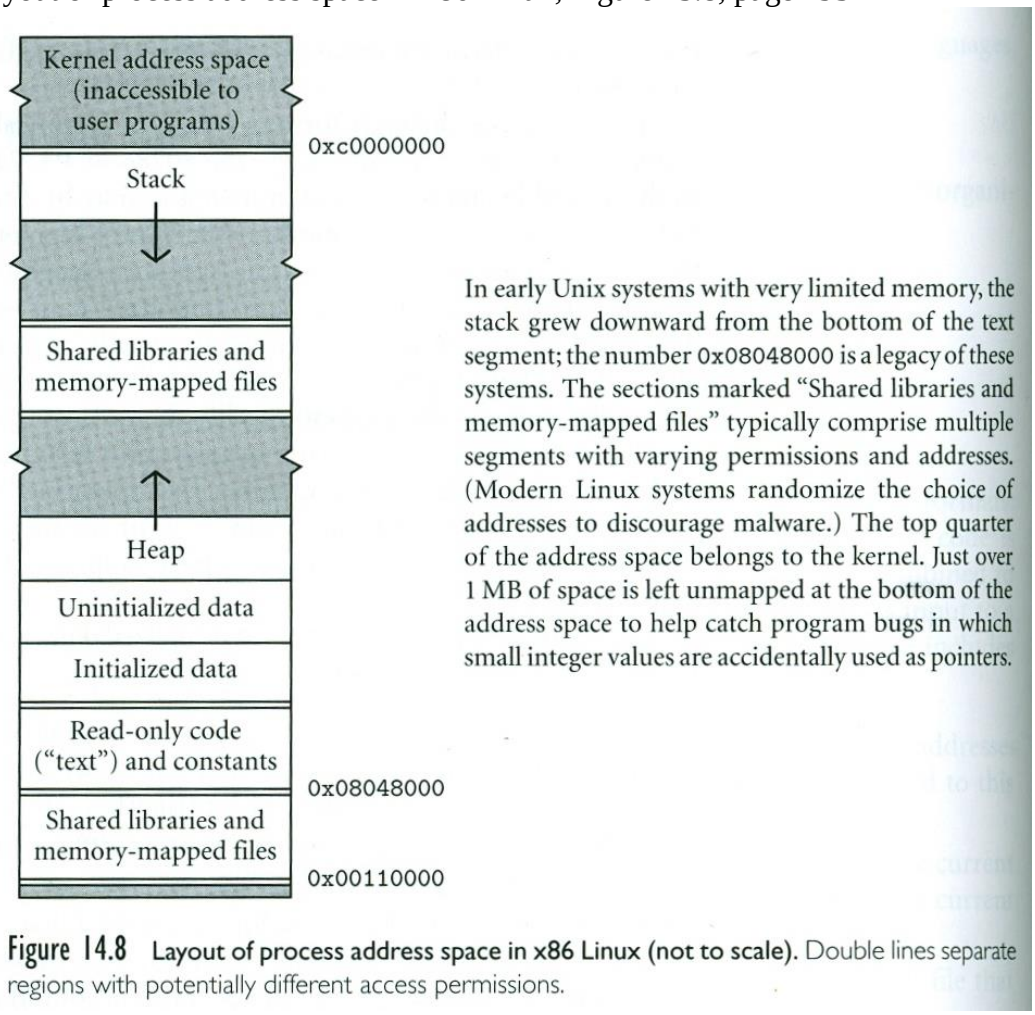

Figure 14.8 Layout of process address space in x86 Linux (not to scale). Double lines separate regions with potentially different access permissions.