Swap space overcommitting allows the system to allocate more memory to processes than is actually available in RAM and swap space. Overcommitting is possible because, typically, each process does not make full use of its allocation. Overcommitting can be controlled on a per-*mmap()* basis using the `MAP_NORESERVE` flag, and on a system-wide basis using `/proc` files.

The *mremap()* system call allows an existing mapping to be resized. The *remap_file_pages()* system call allows the creation of nonlinear file mappings.

Further information

Information about the implementation of *mmap()* on Linux can be found in [Bovet & Cesati, 2005]. Information about the implementation of *mmap()* on other UNIX systems can be found in [McKusick et al., 1996] (BSD), [Goodheart & Cox, 1994] (System V Release 4), and [Vahalia, 1996] (System V Release 4).

49.13 Exercises

49-1. Write a program, analogous to *cp(1)*, that uses *mmap()* and *memcpy()* calls (instead of *read()* or *write()*) to copy a source file to a destination file. (Use *fstat()* to obtain the size of the input file, which can then be used to size the required memory mappings, and use *ftruncate()* to set the size of the output file.)

49-2. Rewrite the programs in Listing 48-2 (`svshm_xfr_writer.c`, page 1003) and Listing 48-3 (`svshm_xfr_reader.c`, page 1005) to use a shared memory mapping instead of System V shared memory.

49-3. Write programs to verify that the `SIGBUS` and `SIGSEGV` signals are delivered in the circumstances described in Section 49.4.3.

49-4. Write a program that uses the `MAP_FIXED` technique described in Section 49.10 to create a nonlinear mapping similar to that shown in Figure 49-5.