Chris Kasper
CS 370-001
Homework Set #1

Chapter 1

2. What is the difference between a system call and a library call?
A system call is a mechanism by which a process can request one of the services from the operating system. They serve as an interface between apps and kernel and are processed in a higher-privilege processor mode. A library call is just an ordinary function call/procedure that is in the app user space. However, a library call can utilize system calls too.

4. Describe how microkernel designs are especially well suited to distributed operating systems.
A microkernel design takes much of the traditional kernel functionality and moves it into other processes managed by the kernel. The idea is to make the kernel smaller and easier to write and maintain. Its main focus is to schedule processes and passing messages among them, this includes to other components of an operating system. This makes them well suited to distributed operating system, which have an objective of coordinating multiple CPUs for processing.

7. What are the security aspects of memory management? Of file system management?
With respect to memory management, it needs to assign areas of memory belonging to processes to areas of physical memory, and also control the sharing of memory (make sure memory in one process can't be accessed by other processes). With file system management, resources need to be protected from unauthorized access (files belonging to one user cannot be altered by another, unless given permissions).

8. Why are special instructions used to implement system calls? Why not use normal subroutine calling instructions?
The special instructions cause the CPU to save the current state of the machine and transfer control to a function in the operating system. This suspends the requesting process, then when the request can be fulfilled, resumes it with the system call results. Subroutine calling instructions do not do this.

9. Virtual machine OSs and systems like Xen are often used to run multiple copies of the same guest OS. What are some advantages of this approach over running a single copy of the guest OS directly on the hardware?
VMs are more portable and allow users to use other operating system (use other computer architectures). Users can allocate resources they want to the VM to utilize, and don't have to interfere with the native OS's file system, besides saving the VM's save state. With these systems, users can simultaneously run more than one OS at a time.

Chapter 3

1. Figure 3-2 doesn't show file system support anywhere. Why not? How are file systems handled in hosted Inferno? In native Inferno?
Inferno extends the UNIX idea that almost everything is a file. Directories and files provided by file servers. These file servers are both built in to the kernel and run as normal applications. Through the use of the file servers, almost every resource and service is provided through an interface connected to a name space and can be used through the usual file operations.

2. Inferno doesn't use a software interrupt instruction to initiate system calls. Would it be feasible to use such an instruction for Inferno running natively? For Inferno running hosted by another OS?
Inferno comes with modules (written in C) that mostly contains the functionality associated with system calls, one of them being the Sys module. Other functionality are features of the Limbo language. These are feasible for running Inferno natively and if running hosted by another OS because the system calls are being run the by the Dis interpreter within Inferno.

3. How many lines of code are contained in each of the directories discussed in Section 3.3.2?
emu: 117403
os: 311513
lib9: 9507
libdraw: 7552
libfreetype: 104890
libinterp: 47956
libkeyring: 1120
libmath: 10934
libmemdraw: 7977
libmemlayer: 1765
libtk: 41007

5. In many time-sharing operating systems (including UNIX), hardware is owned by a superuser or other administrative user. Yet the same people who developed UNIX chose to make such things owned by the logged-in user in Inferno. Why did they use a different approach with Inferno?
They do this depending how Inferno is being operated. Inferno makes use of a special user referred to as "eve", which is known as the host owner. If running native Inferno, "eve" is set to Inferno to act as owner its hardware resources. For hosted Inferno, the user running Inferno becomes "eve".

6. Is it possible that we could fail to open all three initial file descriptors in emuinit( ) but not print the error message? How?
Yes. The fprint call under the if statement tries to utilize file descriptor 2. It is possible that the call can't print to that channel.