CS3104 P3 Report

Approach

I chose to implement a pared-down version of the `stat` syscall, which functions as follows: The user makes the syscall with a flag indicating whether it is requesting information about a directory, a target path, and a buffer of type `stat_result`, which is a struct designed to contain the information about one `fs_node`. N.B. all arguments are passed as integers since they are pointers to userspace memory.
If the `dir_info` flag is set, the syscall searches for the relevant node. If found, it copies the node's name, kind and size (if it is a file) or child count (if it is a directory) into the buffer.
In the event that the `dir_info` flag is not set, the syscall gets the list of children from the relevant `tarfs_node` (by casting the `fs_node`), and copies each one into the buffer provided by the user.

Calls to `ls` first `stat` the provided path, scrutinising the result buffer to see if the target path is a file or directory. If it is a file, then all the information required is already contained within the result buffer and this is printed in short or long form according to the "-l" flag. If the result is a directory, then the result contains the number of children which that directory has, and `stat` is called again, with a new buffer of the correct size to store information about all children, and with the `dir_info` flag set to false. The resulting buffer is then outputted in long or short form again according to the "-l" flag.

Motivation and Limitations

I chose to implement a separate syscall as it reduces the overall complexity of the system, and prevents the user from having to worry about unintended interactions resulting from overloading existing calls. Furthermore, it was not immediately clear to me how the existing syscalls could be overloaded, so I found it easier to start from the ground up.
There are limitations to the reliability of my design. The user can cause a page fault if they provide a buffer which is too small to contain the children of a directory, but it would be quite challenging to implement a system which verifies this, as checking the bounds of a buffer would itself cause a page fault. Furthermore, each call to `ls` requires at most two system calls, but this means that the buffer size can be accurately calculated for each result, removing the possibility of a buffer overflow if a directory has an unexpectedly high number of children. An alternative approach could perhaps be to fix the size of the buffer at some high number n, and only allow `stat` to write n elements to the buffer. This would require only one system call, but the associated overhead of creating a large buffer for the results could outweigh any performance gains seen. Profiling would be required to investigate this further, though, and it would also require limiting directories in the system to only contain n elements, or give a caveat that `stat` is not guaranteed to be accurate for large directories.