**CS3500: Operating Systems**

**Lab 3: System Calls**

21/08/2020

Q1). RISC V *ECALL* instruction is used for User mode to Supervisor/Kernel Mode switch. Which instruction is used to switch privilege mode from Supervisor Mode to Machine Mode?

ECALL always shifts from any mode to Machine Mode. If required certain system calls can be delegated to Supervisor mode. But if this delegation is turned off, then ECALL can be used to switch from Supervisor mode to Machine mode. For example timer interrupts are not delegated to Supervisor mode and hence when a timer interrupt occurs (CLINT instruction) it is handled in Machine mode.

Q2). What is the signature of *kernel/sys\_proc.c/sys\_write()* ? Why is it implemented like that?

uint64 sys\_write(void)

This function does not take any arguments directly because its arguments are present in the user space and it needs to be copied into kernel space. It obtains its arguments from the user space using argfd(), argint() and argaddr(). It obtains a pointer to struct file, an address, and an integer in this manner and it returns an integer signifying the success or failure of the call.

Q3). How are arrays/structures passed to the system call?

Arrays or structures are passed as pointers to the system call. This is because there are only 8 argument registers in RISC V and this may not be sufficient to pass arrays and structures. The data in the structure can then be obtained using fetchaddr().

Q4). What is the use of *kernel/syscall.c/copyout()*?

copyout() is used to copy data from kernel space to user space. This can be used to obtain data specific to the kernel in the user programs. For example it can be used to obtain more information about a process (which is stored in the kernel space in struct proc) in the user space.

Q4). Why is it required to maintain a *trampoline* page as part of both user and kernel address space?

When a trap occurs in user mode, the satp register will be pointing to the user page table which does not map the kernel. Since the trampoline page contains instructions for trap handling, it should be present in the user address space as satp currently points to the user page table. But for the trap to be handled, the satp register should be changed to point to the kernel page table by the instructions in the trampoline page. For the instructions in the trampoline page to continue executing after this change and not crash, we need the trampoline page to be present in the kernel address space at the same virtual address. Hence the trampoline page is part of the kernel address space and the user address space at the same virtual location.

Q5). What is the use of kernel/vm.c/walkaddr()?

This is the function used to obtain the physical address from the corresponding virtual address for user pages. It takes as arguments the page table and a virtual address and it walks the page table to see what the virtual address is mapped to. If the virtual address is larger than the virtual address space or is not mapped or it is mapped but the valid bit / User bit is set to 0 it returns 0 else it returns the corresponding physical address. The valid bit indicates whether the entry is valid and the User bit indicates whether the page is accessible in user mode.

Q6). What is the use of *usys.pl*?

Usys.pl is used to generate the usys.s file which contains the stubs for all the system calls. Basically whenever a syscall is called in the user space the stub code is executed. The stub begins with the label as the name of the system call so that whenever a system call is called it jumps to the stub. The stub then passes the system call number through the a7 register and calls an ECALL which causes the corresponding system call to be handled by the kernel in supervisor mode.

Q7). What method did you use in Problem\_2 as compared to Problem\_1?

For problem 1 I passed a pointer to a character array and used argstr() to obtain system call arguments from the user space. But for problem 2, I passed the number of words and a pointer to an array of character arrays and I first obtained a pointer to the character array containing a word by incrementing in steps of 8 using fetchaddr() and then obtained the character array using fetchstr().

Q8). What is the advantage of having a separate Kernel Stack and User stack?

One of the important jobs of the OS is to isolate the OS from user programs and maintaining separate stacks serves this purpose. This arrangement has two main advantages. First is, this ensures you always have enough stack space for the kernel even if the user stack is full. Second is, this prevents security vulnerabilities as any data left in the kernel space is not accessible to user programs.