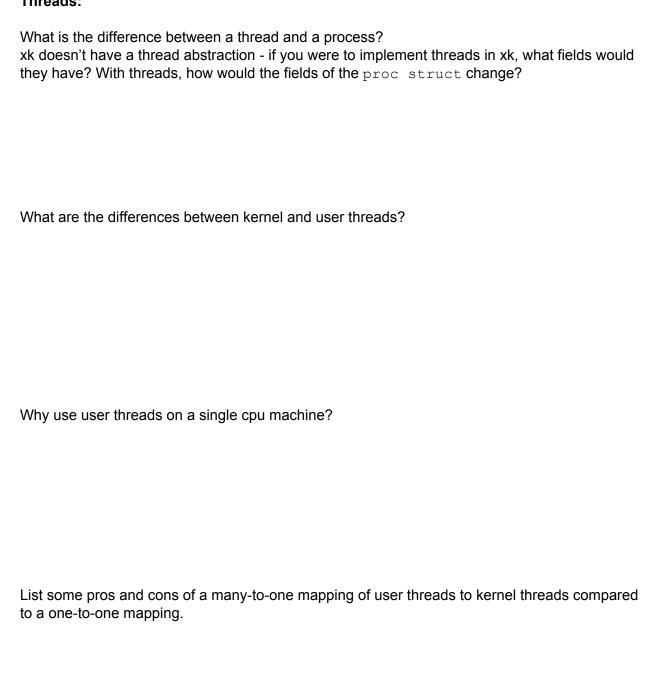
CSE 451: Section 5 Handout

5/2/2019

Threads and Page Faults

Threads:



CSE 451: Section 5 Handout

5/2/2019

Threads and Page Faults

	Page	Faults:
--	------	---------

A trap 14 defines a page fault, this means that the memory address was a not a valid page fo	r
the client to manipulate.	

Can the kernel cause a page fault? If so, how?

For a user process, how will you know if the page fault was caused by attempting to access the stack region of its virtual address space?

Hint: trap.c has a variable addr which is the address the user process tried to access.

The trapframe error code can be read with myproc() ->tf->err.

What will the error code be if the page fault was from touching the stack region of memory?

Can the kernel cause a page fault that was meant for stack growth?

What do the fields of a page (struct vpi) need to be after a copy-on-write fork? Can the kernel cause a copy-on-write page fault?
What will the error code be if the page fault was from touching a copy-on-write page?
When is copy-on-write less efficient than a deep copy fork?