CS61 Lecture 11

October 8, 2019

New Unit: The Kernel

Midterm

- Review monday night
- Start Kernel part before midterm

"It's October it's time for the kernel!!"

Attack Code:

```
push %rbp
mov %rsp, %rbp
jmp 0x... <main+4>
```

```
int main() {
    while (true) { }
}
```

How does the OS not stop when a program infinite loops?

PRIVILEGE: Right to control hardware (disk drive, screen, memory, etc.). Required to access things.

KERNEL: OS software that runs w/ full machine privilege (software that has control over hardware).

Goals:

1. Fairly share machine resources among processes.
2. Provide safe and convenient access to hardware.
3. Robustness and performance.

health of OS remains good

meeting goals doesn't make machine run slowly

PROCESSES: (virtual machines) A program in execution

PS: (linux terminal command ps) that lists all programs in a terminal window

ps axuxww | less : all processes running in system

WeenyOS: small, simplistic OS

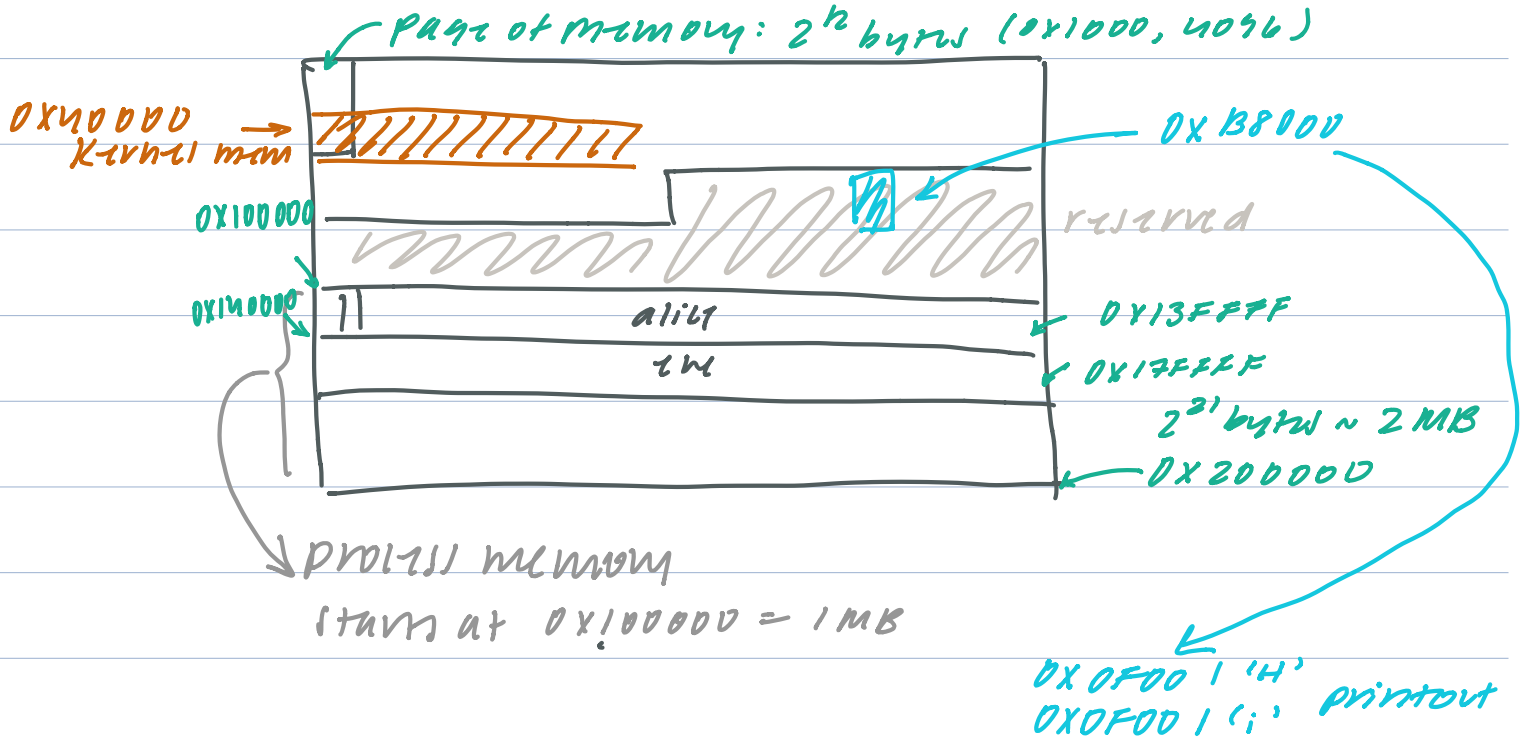
2 processes → WeenyOS → running on machine emulator

process on macOS ← running on VMware ← process on linux

System call: instructions for kernel to execute

sys_yield(): allows something else to run

Primary Memory Management on WeenyOS:



MEMORY MAPPED I/O: Hardware devices controlled through reads and writes to memory.

RESOURCES printer, network, screen, keyboard, mouse, primary memory, CPU time

STARVATION. A process gets zero resource

INTERRUPT. Hardware mechanism that forces kernel execution

TIMER Every time timer goes off, gives control of OS to kernel. → operation it executes must be programmed in

PROGRAMMED I/O. Special instructions for I/O handling

inb/inw/inl
outb/outw/outl } read/write

cli
sti } disable/enable interrupts

Dangerous instructions can change privilege.

CURRENT PRIVILEGE LEVEL.

% CS register: special purpose register

(%CS & 3) → 0 kernel
→ 3 process

*check lower 2 bits of CS register

Modifying CS is something only kernel can do.

attack

① while (true) { }

② asm volatile ("cli")

③ Find system call, overwrite to infinite loop. Attacks kernel memory.

disabled by

① timer interrupt

② changing privilege so processes cannot change interrupts

③

```
vint8_t * x = (vint8_t*) 0x40FA3;
  x[0]
  x[1]
  sys_getpid();
```

①

```
lapicstat::get().ack() → re-enables timer interrupt
schedule() ——————→ runs another process
```

②

```
current -> state = P_BROKEN; → disables current process
schedule();
```