Operating System Design & Implementation

Lab 9: A simple CPU load balancer

TA：周昆霖 吳崧銘

Objective:

In this Lab, you can learn

- The mechanism of CPU load balancing
- Runqueue Structure
- Moving processes between different CPU

Experiment:

- 9-1: Create several processes in running state and dump the runqueue information of each CPU
- 9-2: Move the processes between different CPUs

Exp. 9-1 Dump the runqueue information of all ready processes (including kernel threads)

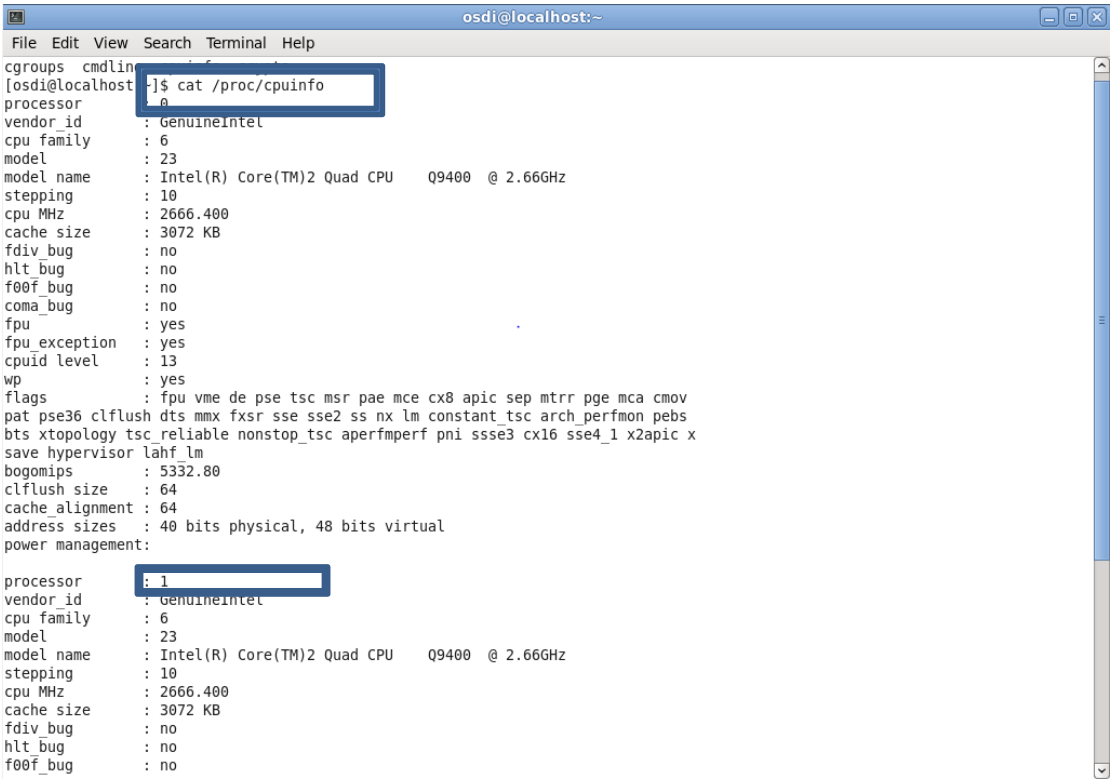
- You are requested to list the runqueue information of all ready processes.

1. Print the number of runnable process in the runqueue list of cpu0 and cpu1
2. Print the currently running process name from the cpu0 and cpu1
   printk("\n the cpu 0 : number of process \n" , ….)
   printk("the name of current process \n" , ….)

   printk("\n the cpu 1 : number of process \n" , ….)
   printk("the name of current process \n" , ….)

Exp. 9-2 Move the ready processes in different scheduling domains

- You are requested to write a simple CPU load (im)balancer and move processes between CPUs (change CPU randomly and periodically for each running process)
- You must make sure you have multiple CPUs in a VM.



- Modify kernel function load_balance located within /usr/src/linux-2.6.32.60/kernel/sched.c

<hint>

if the cpu number is 0){

if the number of process on the runqueue of cpu1 is >= 2{

lock(mutex)

lock(cpu0's runqueue , cpu1's runqueue)

for_each_process{

if the process is user mode{

if process's state is running or ready to run

if process is running at cpu now

do nothing

else

move process from cpu1's runqueue

to cpu0's runqueue

}

}

unlock(mutex)

```
            unlock(cpu0's runqueue , cpu1's runqueue)
        }
return 0;
}
```

Some macro you may use:

for_each_process(p) :traverse all the process

cpu_rq (i): return the address of runqueue of the CPU(i)

task_cpu(p) :get the cpu number

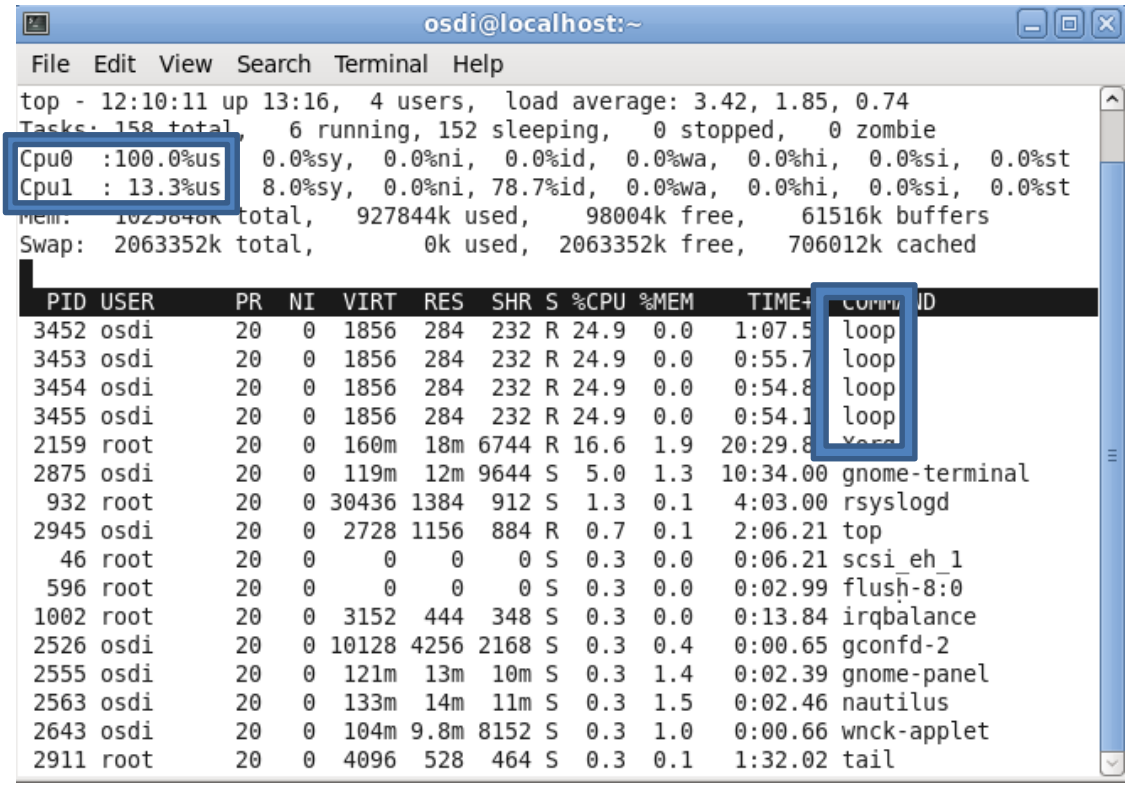
The function you should trace :

Please refer to

/usr/src/linux-2.6-32.60/kernel/sched.c

/usr/src/linux-2.6-32.60/kernel/sched_fair.c

- load_balance
- move_tasks
- load_balance_fair
- __load_balance_fair
- balance_task
- can_migrate_task
- pull_task


osdi@localhost:~
File Edit View Search Terminal Help
top - 12:10:11 up 13:16, 4 users, load average: 3.42, 1.85, 0.74
Tasks: 158 total, 6 running, 152 sleeping, 0 stopped, 0 zombie
Cpu0 :100.0%us 0.0%sy, 0.0%ni, 0.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu1 : 13.3%us 8.0%sy, 0.0%ni, 78.7%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Swap: 2063352k total, 0k used, 2063352k free, 706012k cached
PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
3452 osdi 20 0 1856 284 232 R 24.9 0.0 1:07.5 loop
3453 osdi 20 0 1856 284 232 R 24.9 0.0 0:55.7 loop
3454 osdi 20 0 1856 284 232 R 24.9 0.0 0:54.8 loop
3455 osdi 20 0 1856 284 232 R 24.9 0.0 0:54.1 loop
2159 root 20 0 160m 18m 6744 R 16.6 1.9 20:29.8 Xorg
2875 osdi 20 0 119m 12m 9644 S 5.0 1.3 10:34.00 gnome-terminal
932 root 20 0 30436 1384 912 S 1.3 0.1 4:03.00 rsyslogd
2945 osdi 20 0 2728 1156 884 R 0.7 0.1 2:06.21 top
46 root 20 0 0 0 0 S 0.3 0.0 0:06.21 scsi_eh_1
596 root 20 0 0 0 0 S 0.3 0.0 0:02.99 flush-8:0
1002 root 20 0 3152 444 348 S 0.3 0.0 0:13.84 irqbalance
2526 osdi 20 0 10128 4256 2168 S 0.3 0.4 0:00.65 gconfd-2
2555 osdi 20 0 121m 13m 10m S 0.3 1.4 0:02.39 gnome-panel
2563 osdi 20 0 133m 14m 11m S 0.3 1.5 0:02.46 nautilus
2643 osdi 20 0 104m 9.8m 8152 S 0.3 1.0 0:00.66 wnck-applet
2911 root 20 0 4096 528 464 S 0.3 0.1 1:32.02 tail