

# Cgroups

ภาคการศึกษา 1 2564

กษิต ศาณูเขียว

สาขาวิชาวิทยาการคอมพิวเตอร์

คณะวิทยาศาสตร์และเทคโนโลยี

มหาวิทยาลัย ธรรมศาสตร์

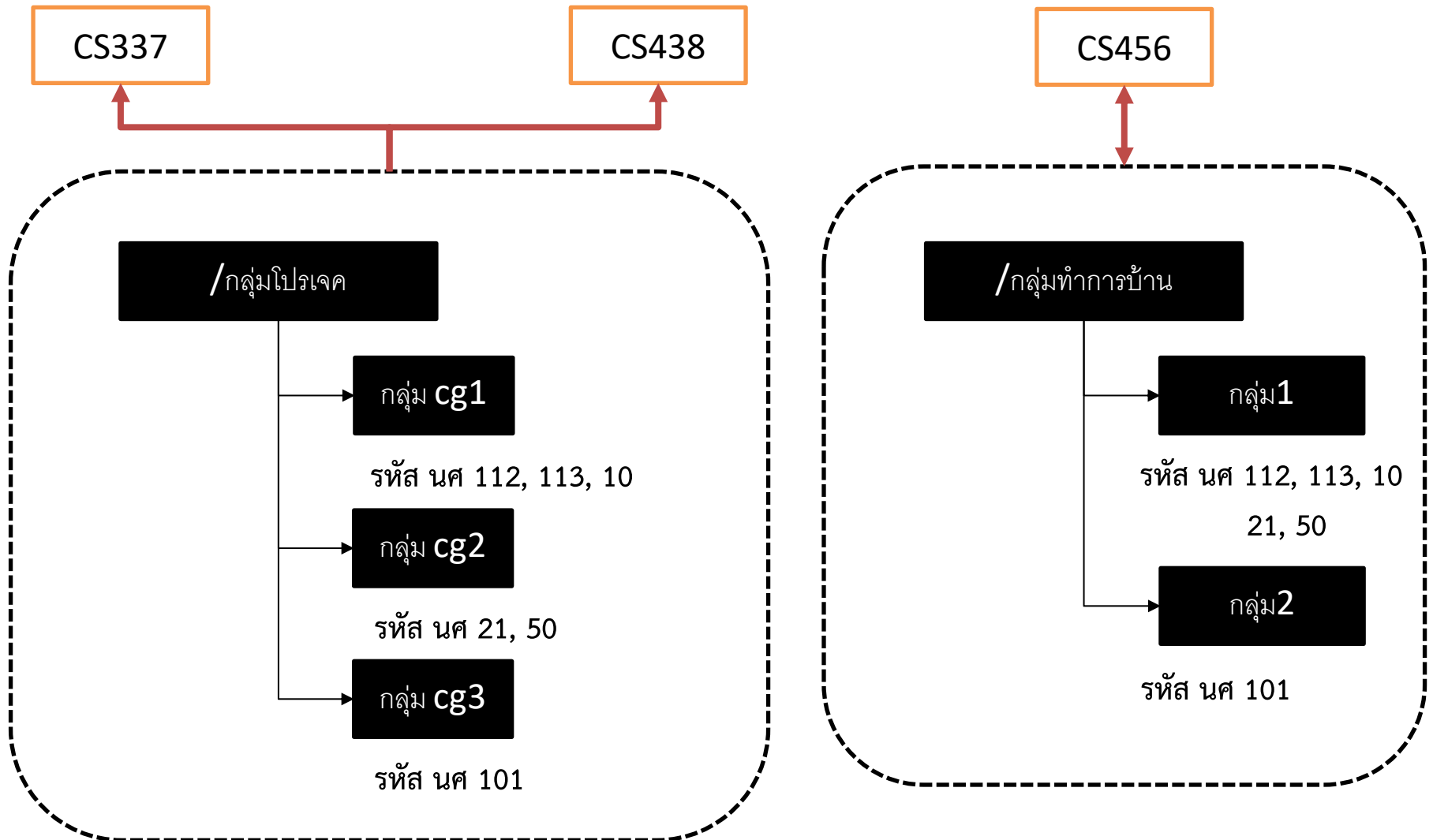
เนื้อหาเรียบเรียงจาก <https://www.youtube.com/watch?v=sK5i-N34im8&t=211s>

# ซีกรูป

- ซีกรูปแบ่งทรัพยากรเป็นระบบย่อย (subsystem) ยกตัวอย่างเช่น ระบบย่อย หน่วยความจำ (Memory Subsystem) ระบบย่อยซีพียู (CPU subsystem) ระบบย่อยบล็อกไอโอ เป็นต้น
- ซีกรูปอนุญาตให้ผู้ใช้สร้างรูปการใช้งานแต่ละระบบย่อย และกำหนดของเขตการใช้งานทรัพยากรในแต่ละระบบย่อยสำหรับแต่ละกรู่นั้น
- ผู้ใช้สามารถ monitor และ configure และ reconfigure และ deny access ของทรัพยากร
- สามารถเซฟและกำหนดค่า configuration ตอน boot time ได้ด้วย
- ทาสค์ (task) หมายถึงโปรเซสในลินุกซ์เคอร์เนล

อ้างอิงจาก [https://access.redhat.com/documentation/en-us/red\\_hat\\_enterprise\\_linux/6/html/resource\\_management\\_guide/ch01](https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/6/html/resource_management_guide/ch01)

เปรียบเทียบ (นศ, กลุ่ม, วิชา) = (โปรเซส, ซีกรูป, ระบบย่อย)



# โครงสร้างต้นไม้ของซีกรูป

- ซีกรูปมีโครงสร้างประกอบไปด้วยโหนดและเอดจ์เชื่อมต่อกันเป็นแผนภาพต้นไม้
- ซีกรูปโหนดมีความสัมพันธ์แบบโหนดพารেন্টและโหนดลูก
- โหนดลูกจะได้รับคุณสมบัติของพารেন্টโหนดเป็นมรดก
- ผู้ใช้สามารถสร้างแผนภาพต้นไม้ของซีกรูปได้หลายแผนภาพ (ที่ไม่เชื่อมต่อกัน)
- แผนภาพต้นไม้จะถูกนำไปแนบ (attach) กับทรัพยากรที่ถูกแบ่งออกเป็นระบบย่อย (subsystem) ดังที่จะได้กล่าวถึงต่อไป

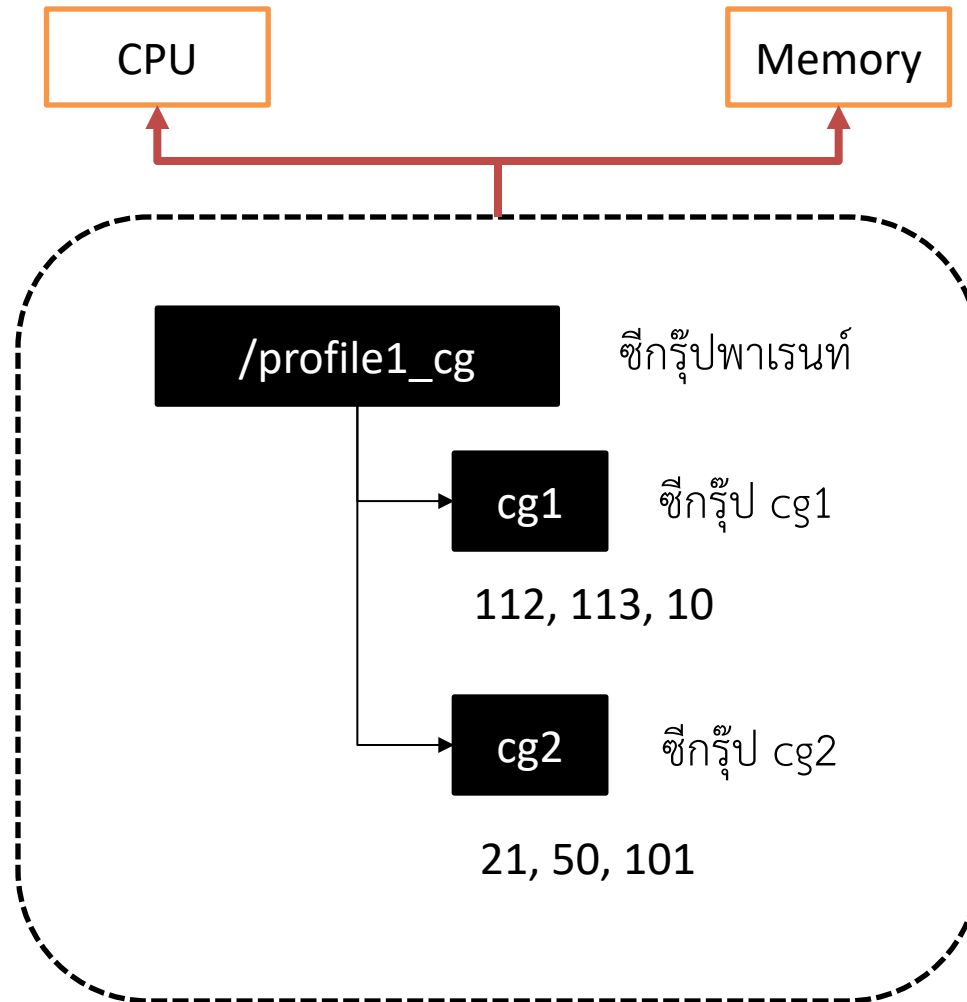
# ระบบย่อย 10 แบบใน RH Linux

- cpu: ให้ scheduler ของลินุกซ์กำหนดการเข้าถึงซีพียูของทาสค์
- cpuacct: สร้างรายงานการใช้งานซีพียูของ(ทาสค์ใน)ซีกรูป
- cpusets: จัดสรรซีพียูคอร์และเมมโมรีให้(ทาสค์ใน)ซีกรูป
- devices: อนุญาตหรือปฏิเสธการเข้าถึงอุปกรณ์ให้(ทาสค์ใน)ซีกรูป
- blkio: กำหนดการจำกัดการเข้าถึงอุปกรณ์บล็อกไอโอ เช่น hdd,sdd,usb
- freezer: หยุดหรือริชুমการประมวลผลของ (ทาสค์ใน)ซีกรูป
- memory: จำกัดการใช้งานหน่วยความจำและรายงานการใช้งานหน่วยความจำของ (ทาสค์ใน)ซีกรูป
- net\_cls: สร้างแทก (tag) สำหรับแพคเกจสำหรับทาสค์ของซีกรูป
- net\_pri: กำหนดไฟอริตี้สำหรับแพคเกจ ต่อ NIC
- ns: เกี่ยวกับเนมสเปซ และ
- perf\_event: เช็คว่าทาสค์อยู่ในซีกรูปไหนและวัดปสม

# กฎความสัมพันธ์ระหว่างชีกรูปและระบบย่อย

1. เราสามารถใช้แผนภาพต้นไม้ชีกรูปแผนภาพหนึ่ง สำหรับกำหนดข้อจำกัดการใช้งานทรัพยากรในระบบย่อยได้มากกว่าหนึ่งระบบย่อย
2. ระบบไม่อนุญาตให้กำหนดให้ระบบย่อยที่มีแผนภาพต้นไม้ชีกรูปแบบอยู่แล้ว ถูกแนบเข้ากับชีกรูปที่มีระบบย่อยอื่นแนบอยู่ด้วย
3. เมื่อเริ่มต้นทาสค์จะเป็นสมาชิกของรูทโหนดของแผนภาพต้นไม้ของชีกรูป ทาสค์จะเป็นสมาชิกของชีกรูปเพียงโหนดเดียวในแผนภาพต้นไม้เดียวกัน
4. เมื่อทาสค์ในชีกรูปสร้างทาว์คลูก ทาว์คลูกจะได้รับมรดกและอยู่ในชีกรูปเดียวกันกับพารেন্ট

# แผนภาพต้นไม้ของซีพียูและหน่วยความจำ



CPU



/cpu

batch

bitcoin

112, 113, 10

realtime

apache

11

redis

21, 50, 101

Memory



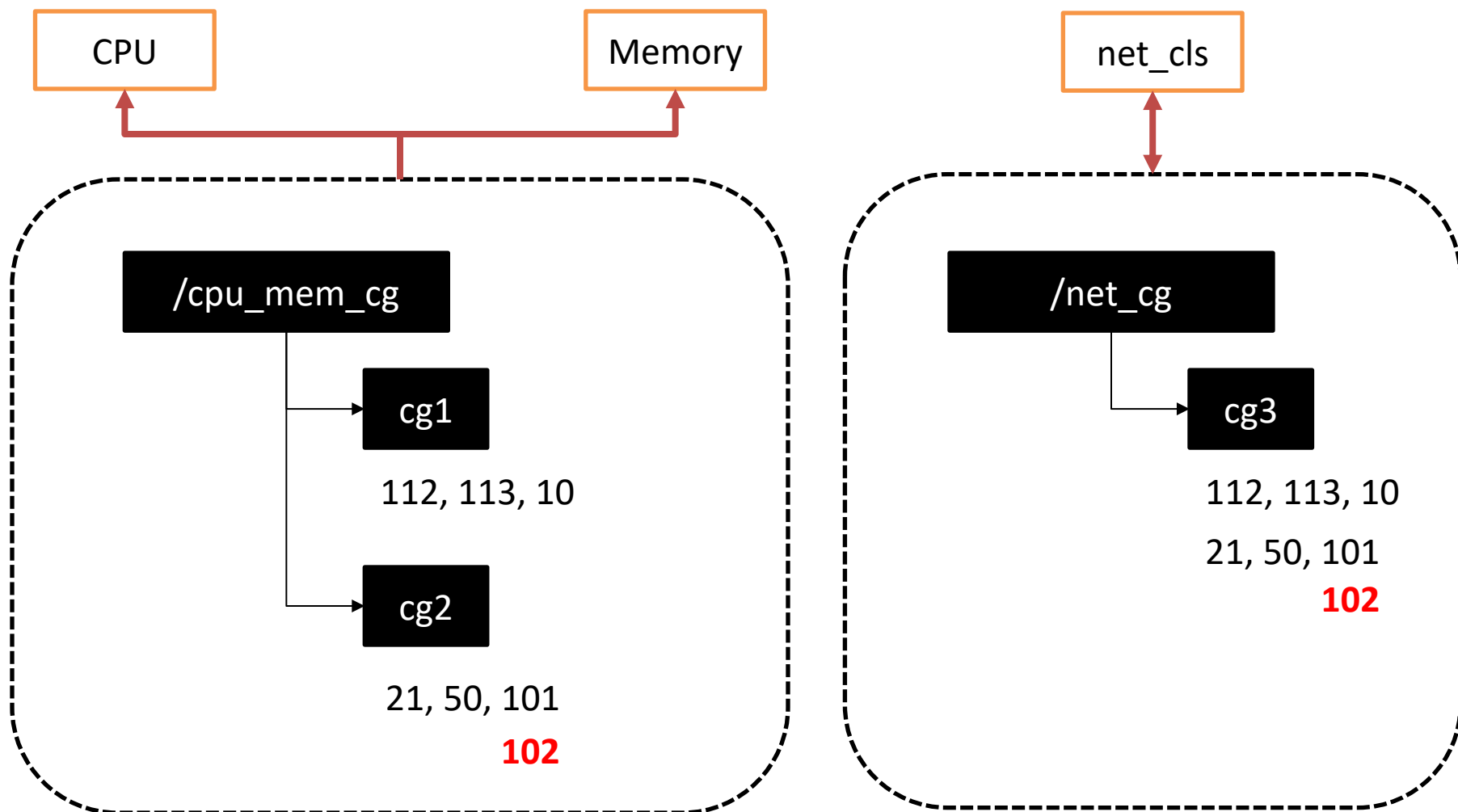
/mem

112, 113,  
10, 11

database

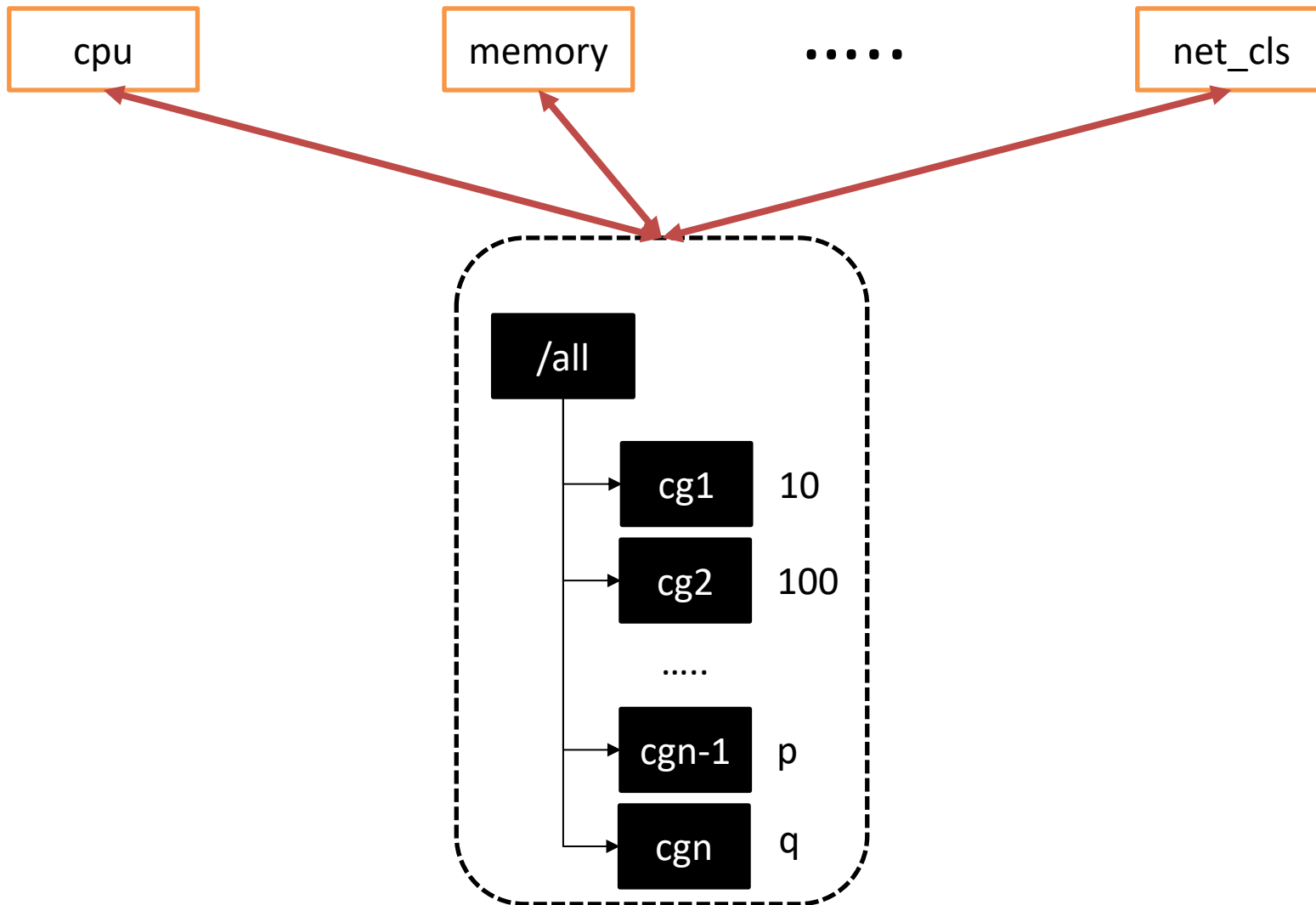
21, 50, 101

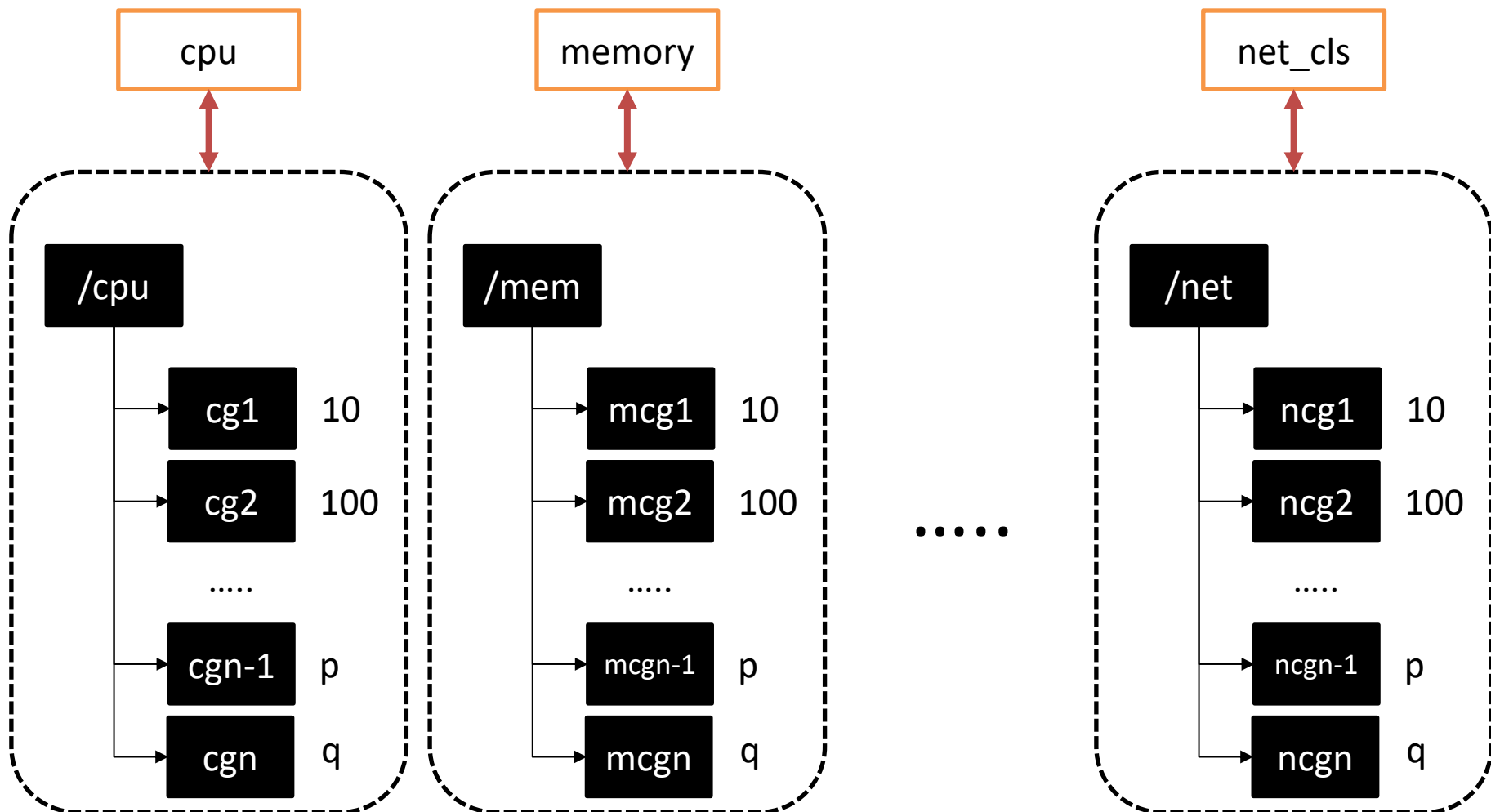




# ผลจากกฎของความสัมพันธ์

- สำหรับแผนภาพต้นไม้ของซีกรูปที่แนบเข้ากับระบบย่อยใดๆ โพรเซสหนึ่ง จะอยู่ภายใต้ข้อจำกัดของซีกรูปเดียวในแผนภาพต้นไม้ที่นั่นเท่านั้น
- เราสามารถแนบแผนภาพต้นไม้เดียวเข้ากับระบบย่อยทุกระบบได้
- บางครั้งเราอาจจำเป็นต้องสร้างแผนภาพต้นไม้ใหม่ และแยกระบบย่อยบางระบบที่แนบกับโครงสร้างต้นไม้เดิมออกไป แล้วแนบโครงสร้างต้นไม้ใหม่เข้ากับระบบย่อยที่แยกออกไปนั้น
- ในกรณีที่ไม่จำเป็นต้องแยกระบบย่อย เราสามารถนำระบบย่อยมาแนบกับโครงสร้างต้นไม้เดิม
- เราสามารถสร้างซีกรูปเฉพาะเจาะจงสำหรับทาส์กลุ่มหนึ่งได้ โดยสร้างโครงสร้างต้นไม้ที่มีซีกรูปโหนดที่มีข้อกำหนดเฉพาะสำหรับทาส์กลุ่มนั้น
- เราสามารถกำหนดให้มีโครงสร้างต้นไม้ที่ทุกโหนดมีโพรเซสเดียวและสร้างแผนภาพต้นไม้แบบเดียวกันสำหรับทุกระบบย่อยได้ ในกรณีนี้เราสามารถกำหนดข้อจำกัดของเฉพาะของโพรเซสทุกโพรเซสโดยแยกเฉพาะสำหรับระบบย่อยแต่ละระบบย่อยได้





# การจัดการซีกรูปของหน่วยความจำจริง

- เคอร์เนลจะนับและจำกัดจำนวนเมมโมรีเพจจริงของเครื่องที่ถูกใช้ในแต่ละกรุป
- เคอร์เนลจะบันทึกเก็บบัญชีว่าเพจไหนอยู่ในกรุปไหน โดยแบ่งชนิดของเพจในสองมิติ มิติแรกคือการแบ่งชนิดของเพจตามลักษณะของข้อมูล
  - เพจจากไฟล์ (file) เป็นเพจที่เกิดจากการโหลดจากไฟล์ด้วย read หรือ mmap อาจเป็นข้อมูลหรือโค้ด จากอุปกรณ์บล็อก เช่น hdd sdd
  - เพจไม่มีชื่อ (Anonymous) เป็นเพจที่ไม่มีเนื้อหาเกี่ยวข้องกับไฟล์เช่นเพจใน heap และ stack segment เป็นต้น
- มิติที่สองคือการใช้งาน
  - แอคทีฟ คือเพจที่ถูกอ้างอิงบ่อย และ
  - อินแอคทีฟเมมโมรีคือเพจที่ไม่ค่อยถูกใช้งาน

## การนับเพจของซีกรูปของหน่วยความจำ

1. เพจจริงหนึ่งๆ (a physical memory page) จะถูกนับว่าเป็นของซีกรูปอินสแตนซ์อันใดอันหนึ่งเท่านั้น
2. ถ้าโปรเซสหนึ่งจบการประมวลผล และเพจที่โปรเซสเคยใช้งานถูกใช้โดยโปรเซสจากกรุปอื่น เพจนั้นจะถูกนับว่าเป็นเพจของกรุปใหม่
3. ในกรณีที่โปรเซสหลายโปรเซสแชร์ข้อมูลจากไฟล์เดียวกัน (เช่นแชร์ไลบรารี) โปรเซสเหล่านั้นอาจมาจากหลายกรุป เพจจึงอาจถูกแชร์โดยหลายกรุปได้ ในกรณีนี้ระบบจะนับว่าเพจที่ถูกแชร์เป็นของกรุปใดกรุปหนึ่งเท่านั้น (เงื่อนไขนี้ทำให้ข้อ 1 เป็นจริง)

# การจำกัดการใช้งานหน่วยความจำในซีกรุป

- ผู้ใช้สามารถกำหนดให้ซีกรุปแต่ละกรุปมีข้อจำกัด (limit) ในการใช้งานหน่วยความจำ
- ผู้ใช้สามารถเลือกที่จะกำหนดข้อจำกัดหรือไม่ก็ได้ โดยดีฟอลต์จะไม่กำหนด ดังนั้นโดยดีฟอลต์โพรเซสในกลุ่มใดๆสามารถใช้หน่วยความจำได้ตามที่ต้องการ
- ในกรณีที่ผู้ใช้ต้องการกำหนดข้อจำกัด สามารถทำได้สองแบบคือ กำหนด ฮาร์ดลิมิต (hard limit) และซอฟต์ลิมิต (soft limit)

# การจำกัดการใช้งานหน่วยความจำในซีกรูป

- ฮาร์ดลิมิต (hard limit) คือการกำหนดค่าขนาดของหน่วยความจำจริงที่ถ้าการใช้งานหน่วยความจำจริงของโปรเซสในซีกรูปเกินขนาดนี้แล้ว เคอร์เนลจะสั่งให้ระบบจัดการการใช้งานหน่วยความจำเกินขนาด (Out Of Memory Killer) หรือระบบ OOM สั่งให้โปรเซสจำนวนหนึ่งในซีกรูปที่ใช้หน่วยความจำเกินจบการประมวลผล
- ซอฟต์ลิมิต (soft limit) ไม่บังคับให้โปรเซสจบการประมวลผลเหมือนฮาร์ดลิมิต แต่จะปล่อยให้โปรเซสในซีกรูปใช้หน่วยความจำเกินขนาดซอฟต์ลิมิตในกรณีที่มีโปรเซสใดจากซีกรูปที่ใช้หน่วยความจำน้อยกว่าซีกรูปนั้นต้องการใช้หน่วยความจำ เคอร์เนลจะนำหน่วยความจำจริงของซีกรูปที่เกินซอฟต์ลิมิตมากที่สุดมาให้กับโปรเซสในซีกรูปที่ใช้หน่วยความจำน้อยกว่า



# การจำกัดการใช้งานหน่วยความจำในซีกรุป

- การจำกัดค่าลิมิตของการใช้งานหน่วยความจำสามารถกำหนดได้สำหรับหน่วยความจำสามแบบ ได้แก่
- หน่วยความจำจริง (physical memory) หมายถึงขนาดการใช้งานหน่วยความจำจริง
- หน่วยความจำจริงสำหรับเคอร์เนล (kernel memory) ขนาดของหน่วยความจำเรสลิเด็นของเคอร์เนลในหน่วยความจำจริงและขนาดหน่วยความจำที่เคอร์เนลใช้ (เพื่อป้องกันไม่ให้เยอะเกินไปซึ่งอาจเป็นผลมาจากการโจมตีเป็นต้น)
- หน่วยความจำจริงรวม (total memory) หมายถึงขนาดการใช้งานหน่วยความจำจริง และพื้นที่ในสวอปสเปซ (swap space)

# ระบบเตือนการใช้งานหน่วยความเกิน OOM Notifier

- แทนที่จะใช้ OOM Killer สำหรับจัดการฮาร์ดลิมิต ผู้ใช้สามารถใช้ระบบเตือนการใช้งานหน่วยความจำเกินฮาร์ดลิมิต (OOM Notifier) แทน
- เมื่อเกิดการใช้งานเกินขึ้น ระบบ OOM Notifier จะทำดังนี้
  1. หยุดการประมวลผลของโพรเซสทั้งหมดในชีกกรุ๊ป
  2. ส่งสัญญาณเตือนให้กับยูเซอร์โปรแกรมที่จัดการทรัพยากร
  3. โปรแกรมหรือผู้ใช้สามารถเลือกที่จะจบโพรเซสบางโพรเซสในชีกกรุ๊ป หรือเพิ่มขนาดของลิมิตของชีกกรุ๊ป หรือไม่เกรตคอนเทนเนอร์ไปรันบนเครื่องอื่น
  4. หลังจากนั้น สั่งให้โพรเซสในชีกกรุ๊ปทั้งหมดประมวลผลต่อ

# การจัดการซีกรูปของหน่วยความจำ

- เมื่อเคอร์เนลให้เพจขอหน่วยความจำจริงให้กับโปรเซสในซีกรูป หรือนำเพจที่โปรเซสเคยใช้ไปให้กับโปรเซสในซีกรูปอื่น เคอร์เนลจะต้องนับจำนวนเพจและทำบัญชีของจำนวนเพจในระบบ ซึ่งทำให้มีโอเวอร์เฮด
- ผู้ใช้ไม่สามารถปิดการนับได้โดยการสั่งโปรเซส แต่ทำได้โดยการกำหนดค่า boot เท่านั้น
- ในกรณีที่เพจถูกใช้โดยโปรเซสจากหลายซีกรูป เคอร์เนลจะนับว่าแชร์เพจนั้นเป็นของซีกรูปแรกที่ใช้ และถ้าซีกรูปแรกเลิกใช้เพจ เพจนั้นจะถูกนับว่าเป็นของซีกรูปถัดไป
- Hugh TLB Cgroup กำหนดปริมาณของ huge pages ที่โปรเซสในซีกรูปสามารถใช้ได้

# ซีพียูซีกรุป

- สามารถนับเวลาการใช้งานซีพียู (โดยแอปและเคอร์เนล) ของโปรเซสในซีกรุป
- สามารถวัดปริมาณการใช้งานซีพียูแต่ละซีพียูโดยโปรเซสในซีกรุป
- อนุญาตให้กำหนดค่าน้ำหนักว่าจะให้โปรเซสในซีกรุปใดมีสิทธิ์ใช้ซีพียูใดได้มากกว่ากัน

# ซีพียูซีกรุป

- แต่ไม่สามารถจำกัดการใช้งานซีพียูของโปรเซสในซีกรุปได้
- เช่น ต้องการให้โปรเซสในซีกรุป A ใช้ซีพียูคอร์ X ได้เพียงแค่ 10% ของความสามารถของซีพียู ไม่สามารถทำได้ เนื่องจากถ้าซีพียูมีการใช้งานน้อย เครื่องอาจลดคล็อกสปีดของซีพียูลงทำให้ซีพียูทำงานช้าลง และโปรเซสในซีกรุปก็จะทำงานช้าลงตามไปด้วย
- หรือ ถ้าต้องการนับจำนวนคล็อกไซเคิล ก็ทำได้ยากเพราะคำสั่งใน ISA ของซีพียูเช่น x86 แต่ละคำสั่งอาจใช้จำนวนคล็อกไซเคิลไม่เท่ากัน และการสร้างไบนารีโค้ดของแต่ละคอมไพเลอร์ก็ไม่เหมือนกันด้วย

# ซีพียูเซต ซีกรูป

- CPUset Cgroups อนุญาตให้ pin โพรเซสในซีกรูปบนซีพียู หรือเซตของซีพียู เพื่อให้รันบนซีพียูที่ pin เท่านั้น
- กำหนดเซตของซีพียูสำหรับแอปใดแอปหนึ่งได้
- เพื่อให้เข้าถึงข้อมูลได้เร็วในกรณีของ NUMA หรือ
- ป้องกันการไม่เกรตข้ามชอกเกตเป็นต้น

# บล็อกไอโอ ซีกรุป (Blkio Cgroup)

- เรียกอีกอย่างว่า Block IO Controller
- เป็นการเก็บบัญชีการใช้งานบล็อกไอโอสำหรับแต่ละซีกรุปว่าแต่ละกรุป
  - ใช้งาน อุปกรณ์บล็อกดีไวส์ (block device) หนึ่งไปเท่าไร
  - อ่านข้อมูลและเขียนข้อมูล กับบล็อกดีไวส์ไปเท่าไร
  - อ่านเขียนข้อมูลแบบ ซิงโครนัส หรือ อะซิงโครนัส ไปเท่าไร
- สามารถ ทrottle (throttling) (หมายถึงการจำกัดแบนวิดท์หรือ จำกัดอัตราการปฏิบัติงาน) สำหรับแต่ละซีกรุป
  - ให้มีอัตราการใช้งานบล็อกดีไวส์หนึ่งๆเท่าไร
  - กำหนดอัตราการอ่านและเขียนข้อมูล (เป็น OPS หรือ Bytes)

# บล็อกไอโอ ซีกรูป (Blkio Cgroup)

- สามารถกำหนดให้ซีกรูปมีน้ำหนักในการใช้งานอุปกรณ์บล็อกดีไวซ์ที่แตกต่างกันได้
- การกำหนดอัตราการเขียนข้อมูลลงสู่บล็อกดีไวซ์ มีสิ่งที่ต้องระลึกถึงเมื่อผู้ใช้อ่านค่าประสิทธิภาพ
  - เช่น ผู้ใช้อาจจะจำกัดค่าแบนวิทสำหรับเขียนข้อมูลลงในอุปกรณ์บล็อกไอโอระดับหนึ่ง แต่ค่าที่วัดได้อาจสูงกว่าที่กำหนดไว้มาก ทั้งนี้เพราะโอเอสอาจเก็บข้อมูลไว้ในหน่วยความจำที่ใช้เป็นหน่วยความจำแคชก่อนแล้วจึงทยอยเขียนลงสู่หน่วยเก็บข้อมูล
  - การเขียนข้อมูลสู่แคชจะทำให้เห็นประสิทธิภาพสูงและไม่ถูกจำกัดตามที่กำหนด แต่การเขียนข้อมูลลงสู่บล็อกดีไวซ์จะเป็นไปตามข้อจำกัดที่กำหนดแต่ผู้ใช้จะไม่เห็น



# เน็ต\_ชีแอลเอส ชีกรูป (net\_cls Cgroup)

- อนุญาตให้ผู้ใช้กำหนดให้ เน็ตเวิร์คแพคเกต (ข้อมูลส่งออก) ที่สร้างโดยโปรเซสในชีกรูป มีแท็กเป็นค่า classid
- ผู้ใช้สามารถใช้ tc utility (traffic control) เพื่อควบคุมแบนด์วิธของข้อมูลที่มีค่าแท็กที่สนใจ
  - `tc class add dev eth0 parent 10: classid 10:1 htb rate 40mbit`
- ผู้ใช้สามารถใช้ iptable เพื่อจัดการแพคเกตที่มีค่าแท็กที่สนใจ
  - `iptables -A OUTPUT -m cgroup ! --cgroup 0x100001 -j DROP`

# เน็ต\_ไพรอร์ ซีกิรूप (net\_prio Cgroup)

- อนุญาตให้แอดมินกำหนดค่าไพอริตี (priority) ของเน็ตเวิร์คทราฟฟิคที่ส่งออกจาก เน็ตเวิร์คอินเตอร์เฟส ให้โปรเซสจากซีกิรूपที่ต่างกันมีค่าไพอริตีสำหรับการส่งออก การเก็บในคิว หรือการยกเลิก ที่ต่างกัน
- การกำหนดค่าไพอริตีของเน็ตเวิร์คทราฟฟิคสามารถทำได้โดยการกำหนดค่า ซอกเก็ตออปชันในแอปพลิเคชัน แต่เรายังจะกำหนดค่าไพอริตีตอนนำแอปมาใช้งานมากกว่า
- แอดมินสามารถใช้ tc utility กำหนดค่าไพอริตีของเน็ตเวิร์คทราฟฟิค ได้เช่นกัน
- `echo "eth0 5" > /cgroup/net_prio/iscsi/net_prio.ifpriomap`

# ดีไวซ์ ซีกรุป (Device Cgroup)

- อนุญาตให้ผู้ใช้กำหนดได้ว่าจะให้โปรเซสในซีกรุปสามารถ อ่าน เขียน หรือ สร้างไฟล์ device ได้
- ผู้ใช้สามารถกำหนดให้โปรเซสในคอนเทนเนอร์ สามารถอ่านเขียนดีไวซ์ที่จะไม่ทำให้เกิดอันตรายต่อทั้งระบบเช่น /dev/tty /dev/random /dev/null และปิดการเข้าถึงอุปกรณ์อื่นที่เหลือ
- ผู้ใช้สามารถกำหนดให้โปรเซสในซีกรุปมองเห็นไฟล์อุปกรณ์ที่สนใจ
  - /dev/kvm : ถ้าต้องการรัน qemu-kvm ในซีกรุป (หรือคอนเทนเนอร์)
  - /dev/net/tun : โปรเซสในซีกรุปเห็นอุปกรณ์ tun ของวีพีเอ็น
  - /dev/fuse : โปรเซสในซีกรุปสามารถสร้างไฟล์ซิสเต็มในยูเซอร์สเปซ
  - /dev/dri : โปรเซสในซีกรุปสามารถใช้ จีพียู

# ฟรีซเซอร์ ชีกรูป (Freezer Cgroup)

- อนุญาตให้ผู้ใช้หยุดการประมวลผลของโปรเซสทั้งหมดในชีกรูป และริซุมการประมวลผล
- คล้ายการส่ง SIGSTOP ซิกแนลไปหยุดโปรเซสเดียวและ SIGCONT ซิกแนลไปริซุมโปรเซส
- สามารถใช้ได้กับการ scheduling โปรเซสทั้งชีกรูป
- ในกรณีที่มีทั้ง client และ server process สามารถหยุดได้ทั้งสองพร้อมๆ กัน เพื่อที่ไม่ให้โปรเซสหนึ่งสังเกตเห็นว่าอีกโปรเซสหนึ่งหยุดและส่งผลกระทบกับโลจิกการประมวลผลของโปรแกรม
- สามารถหยุดทั้งคอนเทนเนอร์แล้ว process migration ไปยังเครื่องอื่นหรือ process checkpointing

# การกำหนดให้โพรセスอยู่ในชีกรูป

- โพรセス PID 1 จะอยู่ที่รูทชีกรูป ของทุกแผนภาพต้นไม้
- เมื่อสร้างโพรセスลูก ลูกจะอยู่ในชีกรูปเดียวกันกับพ่อแม่
- มี utility และไลบรารีสำหรับการจัดการชีกรูปเช่น cgmanager
- เราสามารถใช้ pseudo-fs เพื่อจัดการชีกรูปและการกำหนดให้โพรセスอยู่ภายใต้สังกัดของชีกรูป
  - /sys/fs/cgroup
- pseudo-fs ทำให้สร้าง cgroup ได้ด้วยคำสั่ง mkdir
  - mkdir /sys/fs/cgroup/memory/agroup
- กำหนดหรือให้โพรセスอยู่ใน cgroup หรือ ย้ายโพรセスจากชีกรูปหนึ่งไปยังอีกอันหนึ่ง ได้ด้วยคำสั่ง echo
  - echo \$PID > /sys/fs/cgroup/memory/agroup/tasks

# คอนเทนเนอร์

- คือการประมวลผลแบบเสมือนอย่างเบา (lightweight VM) ที่สร้างภาพเสมือนของการทำงานของคอนเทนเนอร์เหมือนกับเป็นเครื่องจริงเครื่องหนึ่ง ที่ผู้ใช้สามารถ
  - รันแอปพลิเคชัน
  - ประมวลผลเป็นซูเปอร์ยูเซอร์
  - รันเซอร์วิส
  - ติดตั้งแพคเกจ
  - กำหนดค่าเน็ตเวิร์ค เช่นไอพีแอดเดรส
  - มีอุปกรณ์เครือข่าย (เสมือน) เป็นของตนเอง

# คอนเทนเนอร์

- แต่คอนเทนเนอร์ไม่ใช่วีเอ็ม
- คอนเทนเนอร์เป็นกลุ่มของโพรเซสที่รันอยู่บนโฮสโอเอส
- คอนเทนเนอร์สร้างภาพเสมือนของการแยกกลุ่มของโพรเซสออกจากกลุ่มอื่น และแต่ละกลุ่มจะมองไม่เห็นโพรเซสในกลุ่มอื่น
- โฮสโอเอสเห็นโพรเซสในคอนเทนเนอร์ แต่โพรเซสในคอนเทนเนอร์จะเห็นเฉพาะโพรเซสที่อยู่ในคอนเทนเนอร์เดียวกันเท่านั้น
- ไม่สามารถรันต่างโอเอสจากโฮสได้ และไม่สามารถมีเคอร์เนลโมดูลแยกจากของโฮสได้ และทุกคอนเทนเนอร์แชร์เคอร์เนลร่วมกัน
- แต่ละคอนเทนเนอร์มีโพรเซสที่มีพีไอดี 1 แต่ไม่ใช่ init ไม่มี cron

# คอนเทนเนอร์

- คอนเทนเนอร์ เป็นสิ่งที่สร้างขึ้นมาจากส่วนประกอบของเคอร์เนลเรียกว่า ซีกรุป (Cgroup) เนมสเปซ (Namespace) และก๊อปปีออนไรท์ อิมเมจ (Copy On Write) หรือ (Cow)
- ซีกรุป อนุญาตให้เราวัดและกำหนดปริมาณการใช้งานทรัพยากร เช่น หน่วยความจำ ซีพียู บล็อกไอโอ เน็ตเวิร์คไอโอ
- เนมสเปซ อนุญาตให้มีการสร้างอินสแตนซ์ของคุณสมบัติหลายอย่างของเคอร์เนลให้มีมากกว่าหนึ่งอินสแตนซ์ และเคอร์เนลจะแยกอินสแตนซ์เหล่านั้นออกจากกัน
- ก๊อปปีออนไรท์ อิมเมจ อนุญาตให้คอนเทนเนอร์สร้างไฟล์ซิสเต็มอิมเมจของตนเอง และสามารถปรับแต่งอิมเมจตามความต้องการได้





# สรุป

- แนะนำชีקרूपขั้นพื้นฐาน
- ที่เป็นเทคโนโลยีเปิดทางสำหรับสร้างคอนเทนเนอร์

เนื้อหาเรียบเรียงจาก <https://www.youtube.com/watch?v=sK5i-N34im8&t=211s>