Proxy execution

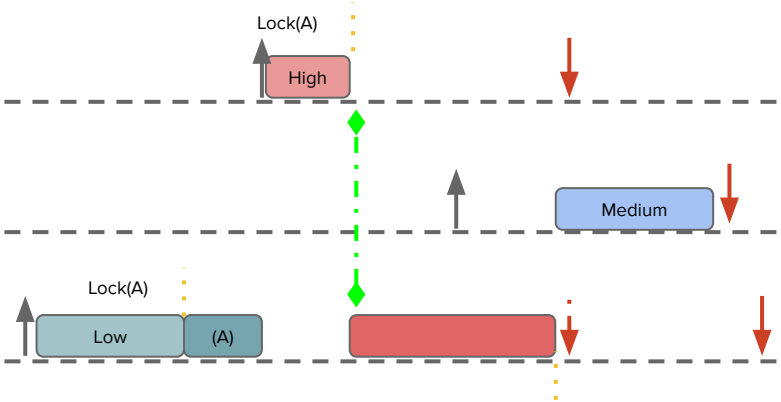
Juri Lelli <juri.lelli@redhat.com>

Linux Plumbers 2019

Red Hat

In a nutshell

- Priority inheritance mechanism
  - Applies to mutexes
  - Replaces (broken) deadline inheritance (SCHED_DEADLINE)
    - Boosted task run outside runtime enforcement (!root prohibited)
  - Works across classes
  - Can unify mutex and rtmutex code
- Mutex owner can run using the scheduling context (“properties”) of (several) donor(s)



migrate_task:

```
/*
 * Follow blocked_on chain.
 */
for (p = next; p->blocked_on; p = owner) {
        mutex = p->blocked_on;
        ...
        owner = __mutex_owner(mutex)
        ...
        if (task_cpu(owner) != task_cpu(p))
                * The blocked-on relation must not cross CPUs, if this happens
                * migrate @p to the @owner's CPU.
                *
                * This is because we must respect the CPU affinity of execution
                * contexts (@owner) but we can ignore affinity for scheduling
                * contexts (@p). So we have to move scheduling contexts towards
                * potential execution contexts.
```

migrate_task:

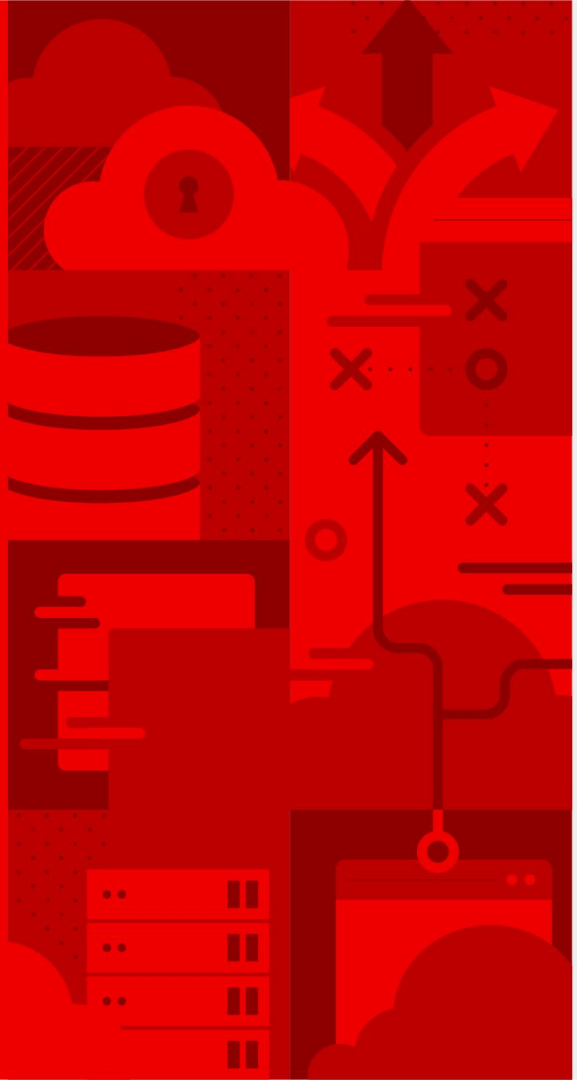
```
/*
 * Follow blocked_on chain.
 */
for (p = next; p->blocked_on; p = owner) {
        mutex = p->blocked_on;
        ...
        owner = __mutex_owner(mutex)
        ...
        if (task_cpu(owner) != task_cpu(p))
                * The blocked-on relation must not cross CPUs, if this happens
                * migrate @p to the @owner's CPU.
                *
                * This is because we must respect the CPU affinity of execution
                * contexts (@owner) but we can ignore affinity for scheduling
                * contexts (@p). So we have to move scheduling contexts towards
                * potential execution contexts.
```

Can we actually do this w/o breaking admission control?

OSPM19 answer… NO! :-(

alternatives

- If p and owner allowed masks are equal -> migrate owner to p's cpu
  - If they were both admitted it means that their bw can be scheduled inside their root domain (DEADLINE doesn't currently care where)
  - What if owner is running? Wait until it is preempted?
  - But then maybe it's actually easier to let p (scheduling ctx) migrate to owner's cpu (as currently implemented)
- If they are disjointed neither p nor owner can be migrated to/from each other domain
  - Don't trigger proxy?
  - Is there theory already for tasks running on separate domains that share data?



Thank you!

Juri Lelli <juri.lelli@redhat.com>

linkedin.com/company/red-hat

facebook.com/redhatinc

youtube.com/user/RedHatVideos

twitter.com/RedHat

Red Hat