memcg priority oom killer

https://github.com/alibaba/cloud-kernel/commit/52e375fcb7a71d62566dc89764ce107e2f6af9ee#diff-8fa1dddd53606ceb933c5c6a12e714ed41e11d37a2b7bc48e91d15b54171d033

在内存压力下，将发生回收和oom. 在一个有多个cgroup的系统中，当有其他候选时，我们可能需要这些cgroup的一些内存或任务在回收和oom中幸存下来。

@memory.low 和 @memory.min已在回收期间发生这种情况，此补丁引入了memcg优先级oom来满足oom中的上述要求。

优先级是从0到12，数字越高优先级越高。 当oom发生时，它总是从低优先级的memcg中选择受害者。它既适用于memcg oom，也适用于全局oom，可以通过 @memory.use_priority_oom 启用/禁用，对于通过**根memcg**的 @memory.use_priority_oom 进行的全局缩放，默认情况下处于禁用状态。

每个mem_cgroup结构体引入了几个和memcg priority的变量

```
@@ -252,6 +255,12 @@ struct mem_cgroup {
        bool            oom_lock;
        int             under_oom;

        /* memcg priority */
        bool use_priority_oom;
        int priority;
        int num_oom_skip;
        struct mem_cgroup *next_reset;

        int     swappiness;
```

原有逻辑也是调用kernel 的 out_of_memory() ，然后调用 select_bad_process 和 oom_kill_process

在原有逻辑中， select_bad_process 阶段，如果是memcg，进行调用memcg自己的函数 mem_cgroup_scan_tasks

新方案，如果oom_control是memcg或者 root_memcg_use_priority_oom() root_memcg 使用 priority_oom，则调用自己实现的 mem_cgroup_select_bad_process(oc);

> 注：所以可能在内存分配上下文(即非memcg的charge阶段)，可能也会调用到memcg的select bad process;

> 而在select中，如果是内存page分配上下文(oc->memcg为空)，
> 则 memcg = root_mem_cgroup ;

如果memcg(可能是当前memcg <在charge上下文> 或root_memcg)使用了 priority_oom ，先调用 mem_cgroup_select_victim_cgroup() 选择一个受害者memcgroup，然后调用之前的 mem_cgroup_scan_tasks 从这个受害者memcgroup中扫描进程(以前方案只有在memcg charge上下文会发生，所以只会当前memcg的扫描task)

> 注:
> 新方案只要开启root_memcg的priority_oom都会调用mem_cgroup的scan_tasks方法？是否合理
> 如果当前memcg没有开启priority_oom，则也不会根据priority选择mem_cgroup

> task_struct->css_set->cgroup_subsys_state->cgroup

在 mem_cgroup_select_victim_cgroup() 中，

1. 如果这个memcg没有hierarchy，则返回当前memcg
2. 获得memcg的subsystem(parent)
3. 获得parent css的memcg(parent_memcg)
4. while(parent)

- 如果parent的task数目小于等于 其对应的memcg不可kill的task数目(num_oom_skip)，跳出循环
- 受害者等于parent
- chosen_priority = 12 + 1 (最高优先级+1)
- 遍历parent subsystem的children(子链表串)css
  - 如果子css的task数目小于等于 其对应的memcg不可kill的task数目(num_oom_skip)，下一个子css
  - 子css的memcg的priority大于chosen_priority，下一个子css
  - 子css的memcg的priority小于chosen_priority，子css优先级更低，遍历子css的子css